НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра автоматики та управління в технічних системах

«На правах рукопису»
УДК 004.855.5

«До захисту допущено»
В. о. завідувача кафедри
_____ О. І. Ролік
«___»_____2018 р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 151 Автоматизація та комп'ютерно-інтегровані технології

на тему: «Методи виявлення образ в коротких текстах»

Виконав:
студент ІІ курсу, групи ІА-61м
Попко Андрій Валентинович _____

Керівник:
Доцент, к.т.н., доцент
Дорогий Я. Ю. _____

Рецензент:
Доц. каф. кібербезпеки та ЗІСТ, к.т.н., доц.
Цуркан В. В. _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2018 року

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра автоматики та управління в технічних системах

Рівень вищої освіти – другий (магістерський) за освітньо-науковою програмою

Спеціальність – 151 Автоматизація та комп'ютерно-інтегровані технології

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ С. Ф. Теленик

«___»_____20__ р.

ЗАВДАННЯ
на магістерську дисертацію студенту

Попку Андрію Валентиновичу

1. Тема дисертації «Методи виявлення образ в коротких текстах», науковий керівник дисертації Дорогий Ярослав Юрійович, доцент, к.т.н., доцент, затверджені наказом по університету від «___»_____ 20__ р. №_____

2. Термін подання студентом дисертації: _____

3. Об'єкт дослідження – методи класифікації текстів.

4. Предмет дослідження – алгоритми машинного навчання в класифікації текстів.

5. Перелік завдань, які потрібно розробити: аналіз існуючих рішень; порівняння базових методів машинного навчання; розробка тестового додатку для дослідження впливу набору характеристик в моделі; визначення шляхів покращення моделі та їх реалізація; аналіз впливу модернізації на якість класифікації.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу: діаграма результатів порівняння точності класифікаторів на основі базових методів машинного навчання; графіки зміни якості моделі в залежності від реалізованих функцій; схема роботи програми.

7. Перелік публікацій: 1) «Дослідження методів визначення забарвленності текстових повідомлень» – Вісник НТУУ «КПІ». Ін-форматика, управління та обчислювальна техніка: збірник наукових праць. – К.: Століття +, 2017. – №66. – С.20-26. (фахове видання); 2) «Визначення образ у коротких текстах» – VI

Міжнародна науково-практична конференція «Summer InfoCom Advanced Solutions 2018» .

8. Дата видачі завдання: _____

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Проведення аналізу існуючих рішень	15.03.18-23.03.18	
2	Порівняння базових методів навчання	24.03.18-29.03.18	
3	Виділення наборів характеристик	30.03.18-09.04.18	
4	Дослідження наборів характеристик	10.04.18-17.04.18	
5	Визначення шляхів покращення моделі	18.04.18-28.04.18	
6	Реалізація функцій покращення	29.04.18-02.05.18	
7	Порівняння з існуючими рішеннями	03.05.18-14.05.18	

Студент А. В. Попко

Науковий керівник дисертації Я. Ю. Дорогий

РЕФЕРАТ

Магістерська дисертація на тему “Методи виявлення образ в коротких текстах”: 132 ст., 23 рис., 25 табл., 2 додатки, 37 джерел.

Об’єктом дослідження виступає процес бінарної класифікації коротких текстів. Предметом дослідження являються методи та засоби штучного інтелекту, що застосовуються в даній сфері.

Метою дисертації являється пошук рішення для можливості бінарної класифікації коротких текстів за наявністю в них образ, а також її дослідження та вибір засобів покращення, що дозволять отримати кращу математичну модель. В ній розглянуто питання вибору математичних засобів для рішення задачі ідентифікації коротких текстів як образливих та реалізації програмного модулю по моделі. Досліджено декілька варіантів моделі. На практиці доведена дієздатність дослідженого методу, що дозволяє отримати високі якісні показники в класифікації.

Методи та підходи, що досліджуються, відносяться до методів машинного навчання по прецедентам. Розроблено додаток на основі математичної моделі на мові Python з використанням бібліотек.

Отримана базова система має потенціал для розвитку та може застосовуватись у сфері адміністрування сайтів, веб-ресурсів, бути частиною інших комерційних продуктів, де необхідно піклуватись про ввічливість спілкування в переписках. На основі результатів було розроблено базовий план для стартап-проекту.

Прогнозні припущення про розвиток дослідження – комбінація з іншими методами машинного навчання та аналізу даних.

КЛАСИФІКАЦІЯ ТЕКСТІВ, МАШИННЕ НАВЧАННЯ, МЕТОДИ МАШИННОГО НАВЧАННЯ, МАШИННЕ НАВЧАННЯ ЗА ПРЕЦЕДЕНТАМИ

ABSTRACT

Master's dissertation on the theme "Methods of revealing image in short texts": 132 pages, 23 figures, 25 tables, 2 appendices, 37 sources.

The object of research is the process of binary classification of short texts. The subject of the study is the methods and means of artificial intelligence used in this field. modernization and search solution for the possibility of binary classification of short texts

The purpose of the dissertation is to find a solution for the possibility of binary classification of short texts on insults presence, its investigation and the choice of means of improvement, which will allow to obtain a better mathematical model. It discusses the choice of mathematical tools for solving the problem of identifying short texts as offensive and implementing a program module by model. Several variants of the model have been explored. In practice, the feasibility of the investigated method is proved, which allows obtaining high qualitative indicators in the classification.

The methods and approaches being studied relate to the methods of machine learning by precedents. The application is developed on the basis of a mathematical model in Python using libraries.

The resulting base system has the potential for development and can be applied in the area of administration of sites, web resources, to be a part of other commercial products, where it is necessary to care about the courtesy of communication in the correspondence. Based on the results, a baseline for the startup project was developed.

Foreseeable assumptions about the development of the study are combination of other methods of machine learning and methods of data analysis.

CLASSIFICATION OF TEXTS, MACHINE LEARNING, MACHINE LEARNING METHODS, MACHINE TRAINING BY PRECEDENTS

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ, ОДИНИЦЬ І ТЕРМІНІВ

VSM – векторна модель представлення тексту

НБА – наївний баєсів алгоритм

SVM – support vector machine, модель навчання з вчителем

LR – Logistic Regression – логістична регресія (модель регресії, де залежна змінна є категорійною)

Data Mining – виявлення прихованих закономірностей або взаємозв'язків між змінними у великих масивах необроблених даних

k-NN – метод k-найближчих сусідів

CSV – Comma-Separated Values, текстовий формат для табличних даних

CV – Перехресна перевірка (Cross-validation), спосіб оцінки точності класифікатора

ВСТУП

Актуальність. Автоматизована класифікація документів чи просто текстів на сьогоднішній день дозволяє заощадити фінансові ресурси та відмовитись від людської участі в даному процесі. За допомогою такого роду класифікаторів можна і розділяти сайти по тематичним каталогам, і боротись зі спамом в електронній пошті, фільтрувати потік новим за темами, тощо. Для рішення таких задач існує достатня кількість підходів та алгоритмів, що показують гарні результати при практичному застосуванні. Проте стандартні підходи не завджи можуть дати гарний результат коли мова піде про операції над короткими текстами. Перш за все, це зумовлено малою кількістю характеристик, що можна добути з таких.

Дана магістерська дисертація присвячена модернізації та пошуку рішення для можливості бінарної класифікації коротких текстів. Її доцільність була обгрунтована прикладною задачею ідентифікації образ в коротких текстах, де в ролі останніх являються коментарі з соціальних мереж та інших веб-ресурсів.

Об’єктом дослідження виступає процес бінарної класифікації коротких текстів.

Предметом дослідження являються методи та засоби штучного інтелекту, що застосовуються в даній сфері та можуть бути покладені в основі класифікатора.

Зв’язок роботи з науковими програмами, планами, темами. Тематика роботи включена в науково-технічні плани кафедри автоматики та управління в технічних системах Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського».

Метою магістерської дисертації є побудова та дослідження математичної моделі для класифікації коротких текстів за наявністю в них образ.

Основною задачею дослідження являється визначення впливу параметрів моделі, введених додаткових характеристик та застосування додаткових функцій обробки тексту на показники якості класифікації. Проміжними задачами можна визначити порівняльний аналіз та огляд існуючих методів машинного навчання, підходів для розв’язання задачі бінарної класифікації, а також вибір засобів

реалізації моделей. В даному дослідженні було використано методи емпіричного дослідження. У вигляді порівнння проаналізовано основні методи машинного навчання, на основі отриманих даних складено різні моделі, на основі яких можна реалізувати класифікатор. Шляхом вимірювання та порівняння оцінено якість кожного. Експерентальних шляхом, на основі аналізу зміни показників якості класифікації підібрані параметри для системи.

Наукова цінність. Одержані результати говорять про високі класифікаційні показники, що не дозволяють отримати стандартні підходи та методи машинного навчання в описаній задачі над короткими текстами. За допомогою модернізацій, описаних в даній магістерській дисертації, вдалось покращити точність класифікації в порівнянні з базовими моделями. Модифікована модель має потенціал для розвитку та може бути вдосконалена в подальшому. Вона являється покращеним варіантом одного з вже існуючих підходів на основі методів машинного навчання та застосовна в умовах малої кількості текстової інформації, що являється одніє з основних проблем в застосуванні готових існуючих рішень.

Практична цінність. Отримана оптимальна модель може бути покладена в основу бінарних класифікаторів коротких тектсів. Останні знаходять своє застосування для автоматизованої інедтифікації повідомлень зі спамом, образ в коментарях соціальних мереж чи просто визначення емоційної забарвленості тектсів.

Апробація результатів дослідження. Результати дослідження апробовані і опубліковані у вигляді тези «Визначення образ у коротких текстах» на VI Міжнародній науково-практичній конференції «Summer InfoCom Advanced Solutions 2018» (додаток А), а також у вигляді статті «Дослідження методів визначення забарвленності текстових повідомлень» у науковому виданні «Вісник НТУУ “КПІ”. Інформатика, управління та обчислювальна техніка №66: збірник наукових праць» (додаток Б).

1 ОСНОВИ ТЕОРІЇ КЛАСИФІКАЦІЇ ТЕКСТІВ ТА ОГЛЯД ІСНУЮЧИХ РІШЕНЬ

Очевидно, що тема даної роботи відноситься до розділу класифікації даних. В ролі таких виступають відносно короткі тексти, що містять певне змістове навантаження. На основі поставлених цілей мова піде про ідентифікацію образ в них. Відповідно рішення для досягнення мети зводиться до визначення способу попередньої обробки текстів, представлення їх в певному вигляді, придатному до відправки для подальшої класифікації.

В цьому розділі буде розглянуто основи теорії обробки тестових даних в площині теорії класифікації.

1.1 Основні етапи класифікації

На основі первинного аналізу та розгляду предметної області було визначено етапи, які неминуче доведеться затронути на шляху досягнення поставлених цілей. Для рішення даної задачі було визначено, що основними являються процеси обробки вхідних даних, який включає певну попередню обробку тексту, виділення з нього певних характеристик, які будуть якимось чином описувати об’єкт (текст); процес безпосередньо класифікації та процес представлення результату. Останній буде залежати від вибраного методу для моделі класифікації.

Для самої ж класифікації існує два способи виконувати подібну класифікацію: з учителем і без вчителя. У методі з учителем модель спочатку тренується на спеціально визначених даних і тільки після цього натреновану модель можна використовувати для присвоювання певних класів з новими даними. Класифікація без вчителя вирішує проблему без тестових розмічених даних і заздалегідь визначених класів, тільки на підставі вмісту самого тексту. Використовуючи такий підхід можна автоматично розподіляти схожі документи по групах. Так, в нашому випадку їх буде всього дві.

Звичайно перед тим як застосовувати алгоритми машинного навчання, дані необхідно перетворити в табличне представлення. Цей процес, представлений на рисунку 1.1 та є найбільш складним і трудомістким.

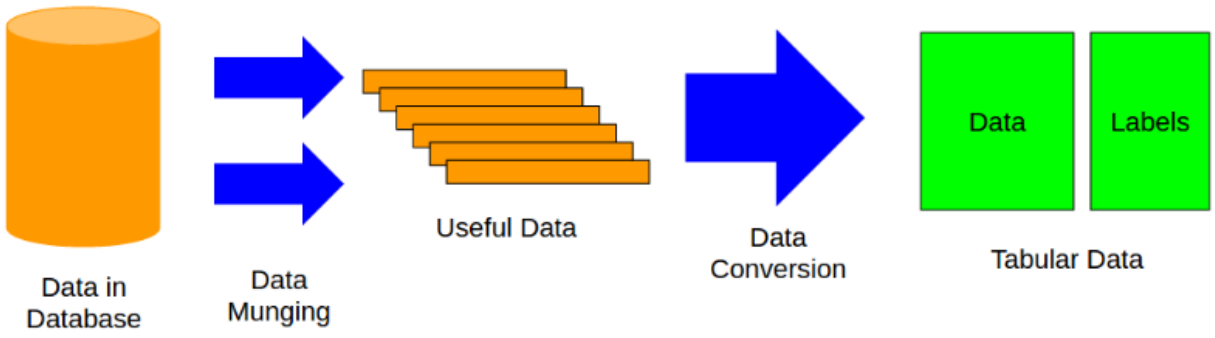

Рисунок 1.1 – Процес отримання даних для навчання

Для вказаного рисунка поясню підписи до елементів: Data in Database - колекція даних, Data Munging - фільтрація даних, Useful Data - корисні дані. Data Conversion - перетворення даних, Tabular Data - табличні дані, де Data - незалежні змінні (ознаки), Labels - залежні змінні (цільові змінні).

Після того, як дані перетворені в табличне представлення, їх можна використовувати для навчання моделей. Табличне представлення є найбільш поширеним представленням даних в сфері машинного навчання та інтелектуального аналізу даних. Рядки таблиці є окремими об'єктами (спостереженнями). Стовпці таблиці, містять незалежні змінні (ознаки), що позначаються X, і залежні (цільові) змінні, що позначаються y. Залежно від класу завдання, цільові змінні можуть бути представлені як одним, так і кількома стовпцями.

1.2 Попередня обробка вхідних даних

Насправді, розуміти просту людську мову для комп'ютера – це досить складне завдання. Занадто багато неоднозначностей, що залежать від контексту. Потрібно справлятися з неологізмами, назвами різних сутностей і ідіомами. Крім

того, для розуміння природної мови, комп'ютеру потрібно мати деяку подобу здорового глузду. Тому, текстові документи представляють як n-розмірний вектор з набором характеристик які описують конкретні об'єкти. Найчастіше це набори слів, які згруповані по унікальності з вказаним значенням повторюваності в документі. При такому підході порядок слів ігнорується і враховується тільки кількість.

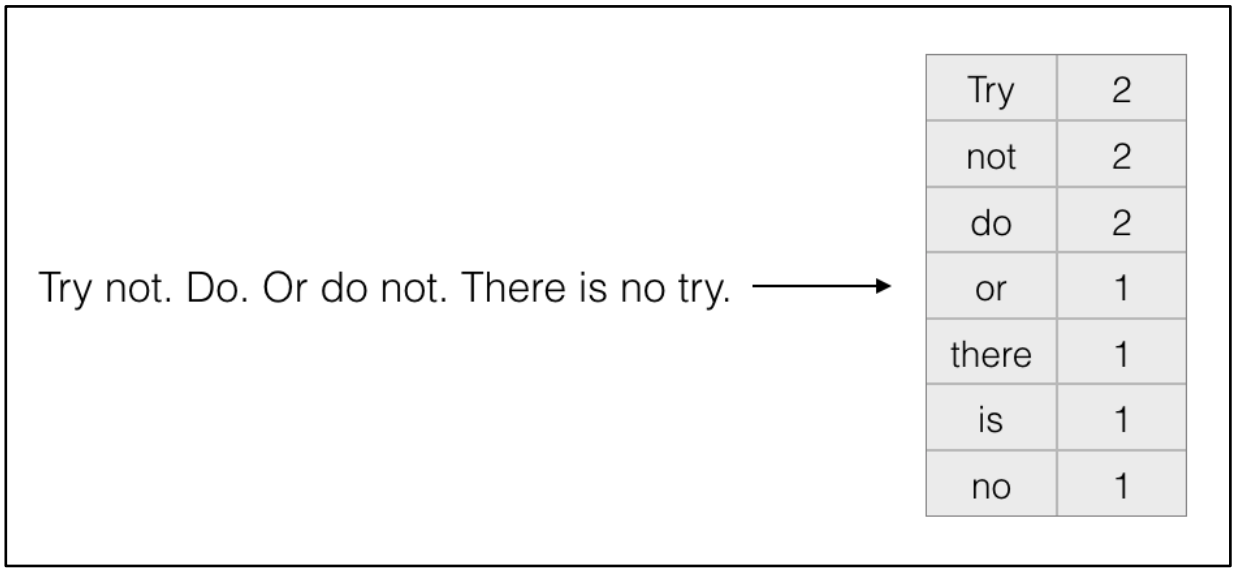

Рисунок 1.2 – Набір слів, що згруповані по унікальності з вказаним значенням повторюваності в документі

Розбивка тексту на окремі слова називається токенізацією (tokenization). Але для підвищення точності потрібно виконати ще ряд кроків. Один з найважливіших кроків - це видалення стоп слів (stop word removal). Нам необхідно видалити слова, що зустрічаються занадто часто, а також слова без значного семантичного сенсу. В англійській мові це «at», «which», «and», «so», «a», «an» і ще багато подібних. Ще потрібно провести нормалізацію всіх слів, виконати стемінг (stemming) і привести всі слова до їх нормальної форми.

Такий підхід дозволить об'єднати різні форми слів, наприклад, слова "працювати", "працював", "працювали" будуть приведені до слова "праця". Тут потрібно врахувати, що для класифікації всі ці слова об’єднуються, і частота буде вважатися як для одного слова.

До речі, основа слова не завжди збігається з морфологічним коренем слова. Можна використовувати лемматизацію (lemmatization). З її допомогою можна знаходити правильні морфологічні основи, що дозволить зберегти зміст слова. Але для лемматизації потрібно використовувати словники, а це значно більш витратний процес, ніж стемінг. І останнє - не варто забувати про приведення всіх слів до одного регістру. На рисунку 1.3 схематично зображено базовий процес обробки вхідного тексту.

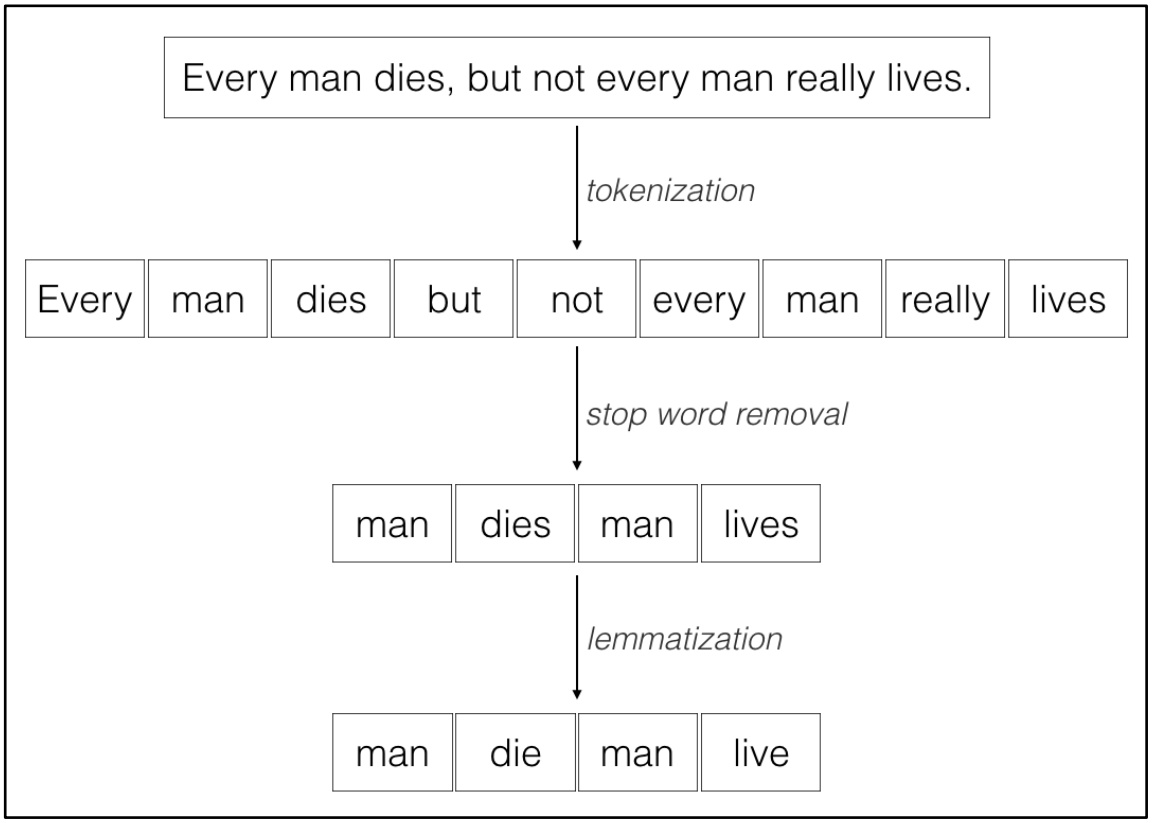

Рисунок 1.3 – Етапи базової обробки вхідного тексту

1.3 Характеристики тексту

Для побудови класифікатора необхідно визначити, які параметри впливають на прийняття рішення про те, до якого класу належить зразок. При цьому можуть виникнути дві проблеми. По-перше, якщо кількість параметрів мала, то може виникнути ситуація, при якій один і той же набір вихідних даних відповідає прикладам, які перебувають в різних класах. Тоді виявиться неможливо навчити

модель, і система не буде коректно працювати (неможливо знайти мінімум, який відповідає такому набору вихідних даних). Вихідні дані обов'язково повинні бути несуперечливі. Для вирішення цієї проблеми необхідно збільшити розмірність простору ознак (кількість компонент вхідного вектора, відповідного зразку).

Але при збільшенні розмірності простору ознак може виникнути ситуація, коли число прикладів може стати недостатнім для навчання системи, і вона замість узагальнення просто запам'ятає приклади з навчальної вибірки і не зможе коректно функціонувати. Таким чином, при визначенні ознак необхідно знайти компроміс з їх кількістю.

Далі необхідно визначити спосіб представлення вхідних даних для нейронної мережі, тобто визначити спосіб нормування. Нормування необхідне, оскільки системи працюють з даними, зазвичай представленими числами в діапазоні 0..1, а вихідні дані можуть мати довільний діапазон або взагалі бути нечисловими даними. При цьому можливі різні способи, починаючи від простого лінійного перетворення в необхідний діапазон і закінчуючи багатовимірним аналізом параметрів і нелінійною нормалізацією в залежності від впливу параметрів один на одного.

Характеристика тексту, як правило, представляє собою деяку величину, яку є можливість обрахувати для даного тексту. Як приклад можна розглядати розподілення довжин слів у виразі. Існує велика кількість задач аналізу тексту, що потребують попереднього визначення його характеристик. До їх числа відноситься, наприклад, задача визначення авторства тексту. Ефективність використання даної характеристики оцінювалась експериментально.

В якості характеристик тексту для визначення його авторства в різні часи пропонувалось використовувати наступні [1–4]:

- Розподіл довжин слів, середня довжина слова [5-7];
- Розподіл сполучень символів деякої довжини (n – буквені n-грами) [8, 9]
- Розподіл частин тексту та їх позиції в реченні. [1-3]. Розглядався розподіл частин на декількох перших позиціях та в кінці речення;

- Розподіл часто вживаних в мові слів. Існують частотні словники, де вказується кількість зустрічань певного слова на мільйон інших в колекції текстів; [8, 9]
- Словарний запас [10, 11]. В роботі [12] також запропоновані показники для тексту як:

$$V_1 = \frac{V}{L}; V_2 = \frac{V}{\sqrt{L}}; V_3 = \frac{\log V}{\log L}; V_4 = \frac{\log V}{\log \log L}, \quad (1.1)$$

де V – число унікальних слів в тексті;

L – їх загальна кількість в тексті.

Пізніше виявилось, що більшість з них підходять і для рішення задачі на визначення емоційної забарвлення та наявність образливих зворотів в тексті.

Існують також дещо інші характеристики, що можна використовувати, якщо мова йде коментарі з соціальних мереж:

- Наявність слів, написаних повністю заголовними літерами[13];
- Наявність слів, що оточені різними астерисками (знаками) [13];
- Наявність смайлів, посилань, хеш тегів тощо [13–15].

1.4 Методи обробки характеристик тексту

Одними із перших методів для визначення емоційного забарвлення тексту, були методи, що засновані на певних правилах (Rule-based methods). Їх робота полягає в пошуку в тексті характерних синтаксичних сполучень, конструкцій, що можуть з високою ймовірність указати на емоційне забарвлення чи наявність образи. Наприклад, вживання частки «ні» перед позитивно забарвленими словами, що характеризують особистість тощо, змінює значення на негативне. Більш складні правила можуть враховувати структуру речення, наприклад, наявність заперечних часток чи сполучників в складносурядних чи складнопідрядних реченнях: «…ти молодець, але (проте)...». Якість таких методів напряму залежить від якості та

кількості запропонованих правил та потребує роботи лінгвіста. Автоматичний синтаксичний розклад речення являється досить складною задачею. Приклад даного підходу розглянуто в роботі [16]. Прості елементи даного підходу використовують разом в сполученні з другими методами, наприклад, машинного навчання.

В деяких випадках в задачах класифікації текстів використовують різноманітні статистичні методи і критерії. В роботі [17] текст розглядають як Марковський ланцюг символів. В роботі [12] запропоновано Хі-квадрат – статистику як міру визначення схожості текстів. Теоретично, такі методи можна використовувати і для рішення задачі визначення образ в коротких текстах (коментарях), проте надійних експериментальних результатів в даному напрямку поки що немає.

Найчастіше на практиці використовують методи машинного навчання для досягнення результату поставленої задачі. На відміну від методів, заснованих на правилах, вони дозволяють враховувати більшу кількість характеристик. Даний підхід дозволяє абстрагуватись від прив’язаності до конкретної мови та являється автоматизованим.

Машинне навчання (Machine Learning) - великий підрозділ штучного інтелекту, що вивчає методи побудови алгоритмів, здатних навчатися. Розрізняють два типи навчання. Навчання по прецедентах, або індуктивне навчання, засноване на виявленні загальних закономірностей по приватним емпіричним даним. Дедуктивне навчання передбачає формалізацію знань експертів і їх перенесення в комп'ютер у вигляді бази знань. Дедуктивне навчання прийнято відносити до області експертних систем, тому терміни машинне навчання і навчання по прецедентах можна вважати синонімами.

Машинне навчання знаходиться на стику математичної статистики, методів оптимізації та класичних математичних дисциплін, але має також і власну специфіку, пов'язану з проблемами обчислювальної ефективності та перенавчання. Багато методів індуктивного навчання розроблялися як альтернатива класичним

статистичним підходам. Багато методів тісно пов'язані з витяганням інформації та інтелектуальним аналізом даних (Data Mining).

Машинне навчання - не тільки математична, а й практична, інженерна дисципліна. Чиста теорія, як правило, не призводить відразу до методів і алгоритмів, які можуть застосовуватися на практиці. Щоб змусити їх добре працювати, доводиться винаходити додаткові евристики, що компенсують невідповідність зроблених в теорії припущень умов реальних завдань. Практично жодне дослідження в машинному навчанні не обходиться без експерименту на модельних або реальних даних, що підтверджує практичну працездатність методу.

Загальна постановка задачі навчання по прецедентах звучить наступним чином. Дано кінцева множина прецедентів (об'єктів, ситуацій), по кожному з яких зібрані (виміряні) деякі дані. Дані про прецедент називають також його описом. Сукупність усіх наявних описів прецедентів називається навчальною вибіркою. Потрібно за цими даними виявити загальні залежності, закономірності, взаємозв'язки, властиві не тільки цій конкретній вибірці, але взагалі всім прецедентам, в тому числі й тим, які ще не спостерігалися. Кажуть також про відновлення залежностей за емпіричними даними.

Найбільш поширеним способом опису прецедентів є ознаковий опис. Фіксується сукупність n показників, вимірюваних у всіх прецедентів. Якщо всі n показників числові, то признакові описи представляють собою числові вектори розмірності n. Можливі й більш складні випадки, коли прецеденти описуються часовими рядами або сигналами, зображеннями, відеорядами, текстами, попарними відносинами подібності або інтенсивності взаємодії, і т. д.

Для розв’язування задачі навчання по прецедентах в першу чергу фіксується модель відновлюваної залежності. Потім вводиться функціонал якості, значення якого показує, наскільки добре модель описує спостережувані дані. Алгоритм навчання (learning algorithm) шукає такий набір параметрів моделі, при якому функціонал якості на заданій навчальній вибірці приймає оптимальне значення. Процес налаштування (fitting) моделі за вибіркою даних в більшості випадків зводиться до застосування чисельних методів оптимізації.

1.5 Векторна модель представлення тексту

Моделювання можна розглядати як заміщення реального об’єкта його умовним еквівалентом, що називають моделлю, що забезпечує близьку до реальної поведінку в рамках допустимих обмежень. Це дозволяє зменшити складність реальних об’єктів, так як модель передає лише необхідні та важливі властивості в даній ситуації.

У більшості випадків ключовими параметрами моделей, що представляють тексти, являються слова, що містяться в тексті. Однією з таких являється векторна модель (VSM – vector space model).

Задача описується наступним чином. Нехай дана колекція текстів $D = (d_1, d_2, \dots, d_m)$, та заданий словник з термінів цієї колекції $T = (t_1, t_2, \dots, t_n)$, . Дана модель представляє кожний текст колекції за допомогою цього словника T як n-мірний вектор, координатами якого являються частоти входження термінів словника в даний текст:

$$\overrightarrow{d_i} = (f_{i1}, f_{i2}, \dots, f_{in}), i = \overrightarrow{1, m} \tag{1.2}$$

В деяких модифікаціях такого підходу частоти входження замінюються вагами, які, по суті, являються тими же частотами, тільки не абсолютними, а відносними. Існує декілька стандартних методик зважування термів.

Однією з таких являється метод TF-IDF [18]. Далі наведені деякі пояснення:

TF (term frequency — частота слова) — відношення числа входжень обраного слова до загальної кількості слів документа. Таким чином, оцінюється важливість слова t_i в межах обраного документа:

$$TF = \frac{n_i}{\sum_k n_k} \tag{1.3}$$

де n_i – число входжень слова в документ;

$\sum_k n_k$ — загальна кількість слів в документі.

IDF (inverse document frequency — обернена частота документа) — інверсія частоти, з якою слово зустрічається в документах колекції [18]. Використання IDF зменшує вагу широковживаних слів.

$$IDF = \log \frac{|D|}{|(d_i \supset t_i)|} \tag{1.4}$$

де $|D|$— кількість документів колекції;

$|(d_i \supset t_i)|$ — кількість документів, в яких зустрічається слово t_i коли $n_i \neq 0$.

Вибір основи логарифму у формулі не має значення, адже зміна основи призведе до зміни ваги кожного слова на постійний множник, тобто вагове співвідношення залишиться незмінним.

Іншими словами, показник TF-IDF це добуток двох множників: TF та IDF:

$$TFIDF = TF * IDF \tag{1.5}$$

Більшу вагу TF-IDF отримають слова з високою частотою появи в межах тексту та низькою частотою вживання в інших документах колекції.

1.6 Моделі класифікації

Первинний аналіз предметної області дає розуміння, що дана задача відноситься до категорії класифікації текстів на основі виділених з них ознак. Одним із способів вирішення таких, коли рішення приймається на основі лінійного оператора над вхідними даними. Він підходить до класу задач, що володіють властивістю лінійної сепарабельності. Кажучи простими словами, операцію лінійної класифікації для двох класів можна представити як відображення об’єктів в багатовимірному просторі на гіперплощину, в якій об’єкти, що попали по одну

сторону розділяючої лінії відносяться до умовно першого класу, а об'єкти по іншу – до другого класу.

Визначення для сказаного формулюється наступним чином. Нехай вектор $\vec{x}$ з дісних чисел представляє собою вхідні дані, а на виході класифікатора визначається показник y за формулою:

$$y = f(\vec{w} \cdot \vec{x}) = f\left(\sum_j (w_j x_j)\right) \quad (1.6)$$

де $\vec{w}$ – дійсний вектор ваг;

f – функція перетворення скалярного добутку.

Другими словами, вектор ваг $\vec{w}$ – коваріантний вектор або лінійна форма відображення $\vec{x}$ в R. Значення ваг для вектору $\vec{w}$ визначається в ході машинного навчання на прикладах. Функція f зазвичай являє собою просту порогову функцію, що відділяє один клас від іншого. В більш складних випадках для неї має місце бути вірогідність для того чи іншого рішення.

На рисунку 1.4 зображено схематично показане відображення об'єктів в просторі, де H_1, H_2, H_3 – гіперплощини, а H_3 – гіперплощина максимальної різниці, тобто така, що розділяє точки на два класи найкращим чином.

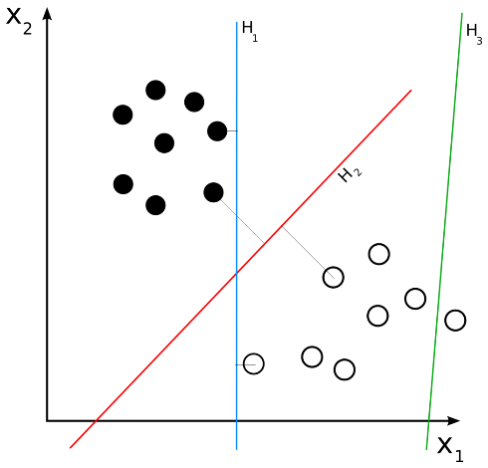

Рисунок 1.4 – Схематичне відображення об'єктів в просторі та їх розділення

Існує два основних підходи для визначення параметрів лінійного класифікатора $\vec{w}$ – породжувальні та розрізнювальні моделі. Перша використовує умовний розподіл $P(\vec{x}|class)$. Сюди можна віднести:

- Лінійний дискримінантний аналіз (або лінійний дискримінант Фішера) (ЛДА, англ. *LDA*) — передбачає ґаусові моделі умовної густини.
- Наївний баєсів класифікатор з поліноміальними або багатовимірними моделями подій Бернуллі.

Розрізнювальні моделі (їх ще називають дискримінативні) прагнуть покращити якість вхідних даних на наборі прикладів для навчання. До таких можна віднести:

- Логістичну регресію – алгоритм, за яким не відбувається передбачення значення числової змінної виходячи з вибірки вхідних значень. Натомість, значенням функції являється вірогідність того, що дане вихідне значення належить до певного класу.
- Простий Перцептрон – алгоритм корекції усіх помилок на вхідному наборі прикладів;
- Метод опорних векторів – алгоритм, який максимізує розділення між гіперплощиною рішення та зразками тренувального набору.

В ході досліджень наявних моделей машинного навчання було визначено що найпопулярнішими являються: Naive Bayes (наївний баєсів класифікатор) [19] , SVM (метод опорних векторів) [20], та Logistic Regression (логістична регресія) [21] для класифікації текстів.

На основі аналізу предметної області було виділено ряд методів, що використовуються для класифікації на основі виділених ознак з тексту.

1.6.1 Лінійні методи класифікації

У галузі машинного навчання метою статистичної класифікації є використання характеристик об'єкту для ідентифікації класу (або групи), до якої він належить.

Лінійний класифікатор (англ. linear classifier) досягає цього ухваленням рішення про класифікацію на основі значення лінійної комбінації цих характеристик. Характеристики об'єкту відомі також як значення ознак, і зазвичай представляються машині у векторі, що називається вектором ознак. Такі класифікатори добре працюють для таких практичних задач, як класифікація документів, і, загальніше, для задач із багатьма змінними (ознаками), досягаючи рівнів точності, порівнянних з нелінійними класифікаторами, у той же час беручи менше часу на тренування та застосування. Далі будуть розглянуті декілька підходів, що відносяться до даної групи.

1.6.1.1 Метод опорних векторів

Даний метод відноситься до категорії алгоритмів навчання з учителем. Він являється одним із найпопулярніших в ній. Запропонований В. Н. Вапником і в англомовній літературі відомий під назвою SVM (Support Vector Machine)[20]. Зазвичай його відносять до бінарних класифікаторів, хоча існують випадки, коли його адаптують і змушують працювати для мультикласифікації. Особливою властивістю методу опорних векторів являється неперервне зменшення емпіричної помилки класифікації та збільшення зазору, тому його також називають як «метод класифікатора з максимальним зазором».

Основна ідея полягає в переводі векторів до простору більш великої розмірності та пошуку розділяючої гіперплощини з максимальним зазором в даному просторі. Дві паралельні гіперплощини будуються по обидві сторони гіперплощини, що розділяє класи. В ролі останньої буде така гіперплощина, що максимізує відстань до двох паралельних гіперплощин. Алгоритм працює в припущенні, що чим більша різниця між ними, тим менша буде середня похибка класифікатора. Формальне описання методу та його застосування наведено далі.

Нехай $Y = \{1, -1\}$. Метод опорних векторів будує класифікатор наступного вигляду:

$$a(x, w, w_0) = \text{sign}\left(\sum_{i=1}^{n} w_i \xi_i - w_0\right) \qquad (1.7)$$

де $w = (w_1, \ldots, w_n) \in \Re^n$, $w_0 \in \Re$ – параметри алгоритму. Рівняння $\langle w, x\rangle = w_0$ описує гіперповерхню в просторі $\Re^n$. Ідея методу заключається в побудові оптимальною в деякому розумінні гіперповерхні, що буде розділяти класи $y = 1$ та $y = -1$. Вибірка лінійно подільна, якшо існує така гіперповерхня, що об’єкти різних класів находяться по різні сторони даної поверхні. Оптимальною називається така розділяюча гіперповерхня, що відстань від неї до найближчого об’єкта максимальна у порівнянні с іншими розділяючими поверхнями. Ці вимоги записуються системою як:

$$\begin{cases} \langle w, w\rangle \to \min; \\ y_i(\langle w, x_i\rangle - w_0) \geqslant 1, \;\; i = 1 \ldots l. \end{cases} \qquad (1.8)$$

Вибірки на практиці зазвичай являються лінійно нероздільні. Метод узагальнюється на цей випадок шляхом дозволу похибок на об'єктах. В такому випадку вимоги будуть записуватись системою як:

$$\begin{cases} 0.5\langle w, w\rangle + C\sum_{i=1}^{k} e_i \to \min; \\ y_i(\langle w, x_i\rangle - w_0) \geqslant 1 - e_i, \;\; i = 1 \ldots l; \\ e_i \geqslant 0, \;\; i = 1 \ldots l. \end{cases} \qquad (1.9)$$

Тут e_i – розмір похибки на об’єкті x_i, C – параметр алгоритму, оптимальне значення якого знаходиться експерементально і регулює співвідношення між шириною смуги і сумарною похибкою на об’єктах.

У випадку коли вибірка лінійно неподільна зазвичай використовується «ядровий трюк» (*kernel trick*). Його суть наступна. Нехай дано відображення $\varphi: \Re^n \to H$, де H – простір із скалярним добутком $\langle a_1, a_2\rangle_H$. Якщо цей простір має більш високу розмірність, то вельми ймовірно, що вибірка в ньому буде роздільна.

Рівняння розділяючої гіперповерхні в просторі H буде мати вигляд $\langle \varphi(X), w \rangle H = w_0$. Відповідь на питання «Якою буде поверхня та як буде виглядіти в просторі» залежить від H. На практиці беспесередньо відображення нас не цікавить, достатньо знати ядро – функцію $K(x_1, x_2) = \langle \varphi(x_1), \varphi(x_2) \rangle_H$.

Приклади ядер:

- $K(x_1, x_2) = \langle (x_1), (x_2) \rangle_{\Re^n}$ – лінійне ядро, розділяюча поверхня матиме вид гіперповерхні;
- $K(x_1, x_2) = (\langle (x_1), (x_2) \rangle_{\Re^n})^d$ – поліноміальне ядро, розділяюча поверхня буде мати вид поверхні порядку d;
- $K(x_1, x_2) = \text{e}^{\frac{|x_1 - x_2|^2}{2\sigma^2}}$ – гауссове ядро с параметром σ;
- $K(x_1, x_2) = th(k_0 + k_1 \langle (x_1), (x_2) \rangle_{\Re^n})$ – сигмоїдне ядро с параметрами k_0, k_1.

Існують правила, за якими можемо побудувати свої ядра, проте в більшості випадків на практиці використовують одні з вказаних вище ядер. Для модифікації алгоритму с урахуванням ядер достатньо всюди в програмі зробити заміну звичайного скалярного добутку на функцію ядра.

Даний метод опорних векторів широко використовують в різноманітних задачах машинного навчання, в яких він показує гарні результати. Можливість варіювати ядро K и параметр C забезпечує методи необхідною гнучкістю. В загальному випадку тимчасова складність алгоритму достатньо висока, а саме поліноміальна від вибірки. Для лінійного ядра існують методи, що дозволяють найти близьку до оптимальної гіперповерхню за лінійний проміжок часу.

Якщо тренувальні дані є лінійно роздільними, то ми можемо обрати дві паралельні гіперплощини, які розділяють два класи даних таким чином, що відстань між ними є якомога більшою. Область, обмежена цими двома гіперплощинами, називається «розділенням» (англ. "margin"), а максимально роздільна гіперплощина є гіперплощиною, яка лежить посередині між цими двома.

На рисунку 1.5 зображена максимальна розділова гіперплощина та межі, натренованої зразками з двох класів. Зразки на межах називаються опорними векторами.

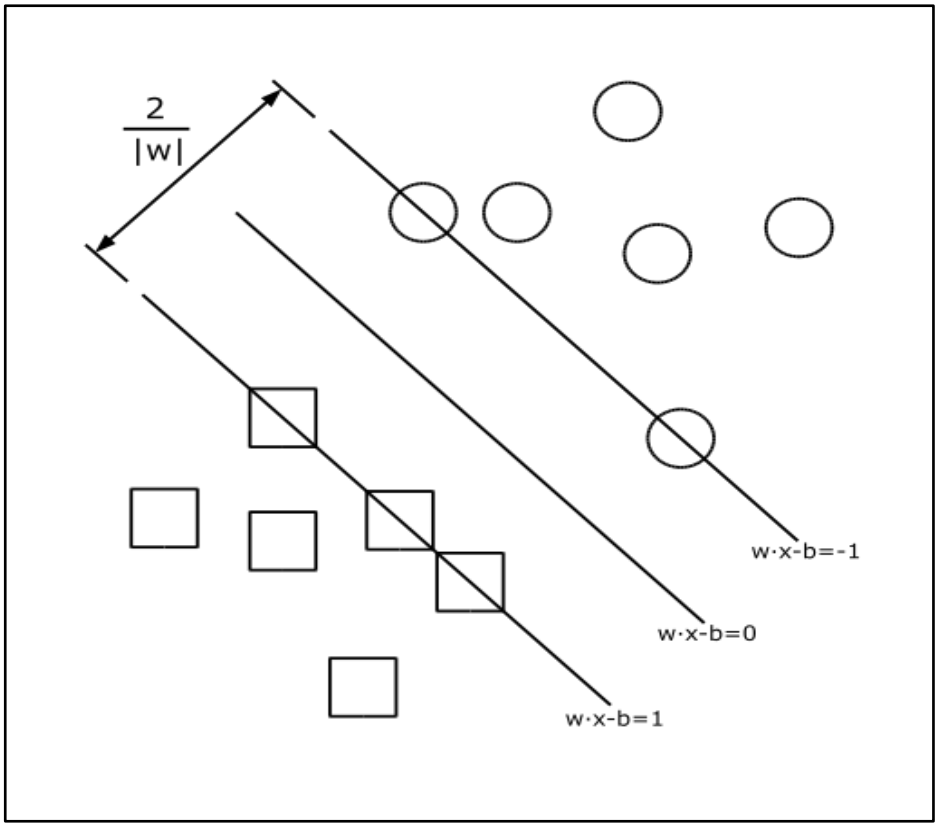

Рисунок 1.5 – Оптимальна розділяюча гіперплощина для методу SVM

Даний метод являється одним з найпопулярніших в задачах бінарної класифікації та показує гарні результати.

1.6.1.2 Логістична регресія

Логістична регресія (Logistic regression) – метод побудови лінійного класифікатора, що дозволяє оцінювати апостеріорні ймовірності приналежності об'єктів класів.

Логістична регресія - це різновид множинної регресії, загальне призначення якої полягає в аналізі зв'язку між декількома незалежними змінними (також званими регресорами або предикторами) і залежною змінною. Бінарна логістична регресія, як випливає з назви, застосовується в разі, коли залежна змінна є бінарною

(тобто може приймати тільки два значення). Іншими словами, за допомогою логістичної регресії можна оцінювати вірогідність того, що подія настане для конкретного випробуваного (хворий/здоровий, повернення кредиту/дефолт і т.д.).

Як відомо, регресивні моделі можуть бути записані у вигляді формули:

$$y = F(x_1, x_2, \dots, x_n) \tag{1.10}$$

Наприклад, в множинній лінійній регресії передбачається, що залежна змінна є лінійною функцією незалежних змінних, тобто:

$$y = a + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n \tag{1.11}$$

Чи можна її використовувати для завдання оцінки ймовірності результату події? Так, можна, обчисливши стандартні коефіцієнти регресії. Наприклад, якщо розглядається варіант визначення чи являється короткий текст образливим, чи ні, задається змінна y зі значеннями 1 і 0, де 1 означає позитивну відбовідь (текст являється образливим), а 0, що поданий текст – нейтральний. Однак тут виникає проблема: множинна регресія не «знає», що змінна відгуку бінарна за своєю природою. Це неминуче призведе до моделі з прогнозуючими значеннями більшими за 1 і меншими 0. Але такі значення взагалі не припустимі для початкової задачі. Таким чином, множинна регресія просто ігнорує обмеження на діапазон значень для y.

Для вирішення проблеми завдання регресії може бути сформульована інакше: замість передбачення бінарної змінної, ми будемо прогнозувати безперервну змінну зі значеннями на відрізку [0,1] при будь-яких значеннях незалежних змінних.

Це досягається застосуванням наступного регресійного рівняння (логіт-перетворення):

$$p = \frac{1}{1 + e^{-y}}, \tag{1.12}$$

де p - ймовірність того, що відбудеться подія, що нас цікавить;

e - основа натуральних логарифмів 2,71;

y - стандартне рівняння регресії.

Формула вище (формула 1.12) називається також логістичною функцією (сигмоїдом), графік якої представлений на рисунку 1.6.

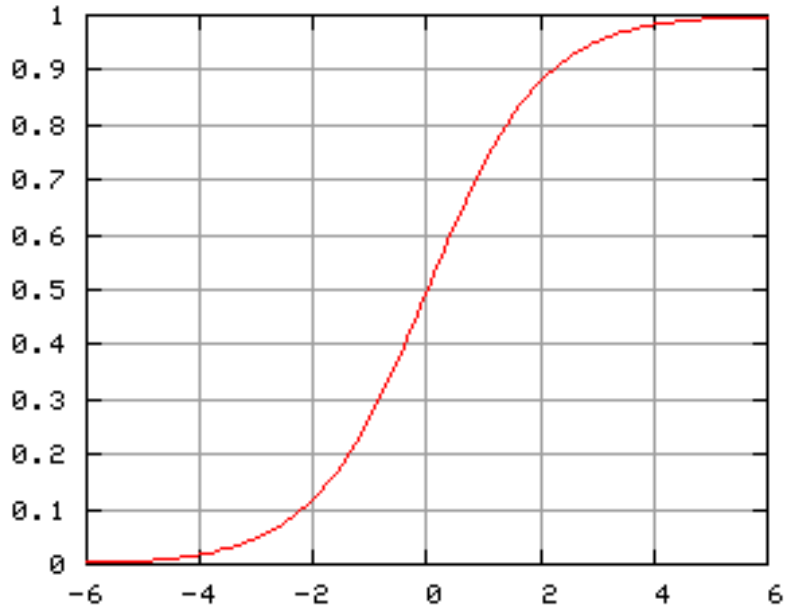

Рисунок 1.6 – Графік логістичної функції

В задачах класифікації за допомогою логістичної регресії робиться припущення, що вірогідність настання події $p = 1$ рівна:

$$\mathbb{P}\{y = 1 \mid x\} = f(z), \tag{1.13}$$

де z= $\theta^T x = \theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n$;

x, θ – вектори-стовбці значення незалежних змінних $1, x_1, x_2, \dots, x_n$ та параметрів (коефіцієнтів регресії) – дійсних чисел $\theta_0, \dots, \theta_n$, відповідно, а $f(z)$ - так звана логістична функція (її також називають сигмоїдом або логіт-функцією):

$$f(z) = \frac{1}{1 + e^{-z}} \tag{1.14}$$

Так як y приймає лише значення 0 та 1, то вірогідність першого можливого значення рівна:

$$\mathbb{P}\{y = 0 \mid x\} = 1 - f(z) = 1 - f(\theta^T x) \tag{1.15}$$

Для стислості функцію розподілу p при заданому x можна записати в наступному вигляді:

$$\mathbb{P}\{y \mid x\} = f(\theta^T x)^p \left(1 - f(\theta^T x)\right)^{1-p}, y \in \{0,1\} \tag{1.16}$$

Фактично це є розподілом Бернулі з параметром, рівним $f(\theta^T x)$.

Для підбору параметрів $\theta_0, \dots, \theta_n$, необхідно скласти навчальну вибірку, що складається з наборів значень незалежних змінних і відповідних їм значень залежної змінної y. Формально, це множина пар $\left(x^{(1)}, y^{(2)}\right), \dots, \left(x^{(m)}, y^{(m)}\right)$, де $x^{(i)} \in R^n$ – вектор значення незалежних змінних, а $y^{(i)} \in \{0, 1\}$ – відповідне їм значення y. Кожна така пара називається навчальним прикладом.

Зазвичай використовується метод максимальної правдоподібності, згідно з яким вибираються параметри θ, що максимізують значення функції правдоподібності на навчальній вибірці:

$$\hat{\theta} = argmax_{\theta} L(\theta) = argmax_{\theta} \prod_{k=1}^{n} \mathbb{P}\{y = y^{(i)} \mid x = x^{(i)}\} \qquad (1.17)$$

Максимізація функції правдоподібності еквівалентна максимізації її логарифма:

$$\ln L(\theta) = \sum_{i=1}^{m} \log \mathbb{P}\{y = y^{(i)} \mid x = x^{(i)}\} = \\ = \sum_{i=1}^{m} y^{(i)} \ln f(\theta^T x^{(i)}) + \left(1 - y^{(i)}\right) \ln(1 - f(\theta^T x^{(i)})), \qquad (1.18)$$

де $\theta^T x^{(i)} = \theta_0 + \theta_1 x_1^{(i)} + \cdots + \theta_n x_n^{(i)}$.

Для максимізації цієї функції може бути застосований, наприклад, метод градієнтного спуску. На практиці також застосовують метод Ньютона та стохастичний градієнтний спуск.

Для поліпшення узагальнюючої здатності отриманої моделі, тобто зменшення ефекту перенавчання, на практиці часто розглядається логістична регресія з регуляризацією.

Регуляризація полягає в тому, що вектор параметрів θ розглядається як випадковий вектор з деякою заданою апріорної щільністю розподілу $p(\theta)$. Для навчання моделі замість методу найбільшої правдоподібності при цьому використовується метод максимізації апостеріорної оцінки, тобто шукаються параметри θ, максимізуючі величину:

$$\prod_{k=1}^{n} \mathbb{P}\{y^{(i)} \mid x^{(i)}, \theta\} * p(\theta) \qquad (1.19)$$

Ця модель часто застосовується для вирішення завдань класифікації - об'єкт x можна віднести до класу $y = 1$, якщо передбачена моделлю ймовірність $P\{y =$

$1 \mid x\} > 0,5$ і до класу $y = 0$ в іншому випадку. Отримувані при цьому правила класифікації є лінійними класифікаторами.

1.6.2 Баєсові методи класифікації

Баєсів підхід до класифікації заснований на теоремі, яка стверджує, що коли щільність розподілу кожного з класів відома, то шуканий алгоритм можна виписати в явному аналітичному вигляді. Більше того, даний алгоритм оптимальний, тобто володіє мінімальною вірогідністю помилок.

На практиці щільності розподілу класів, як правило, не відомі, тож їх доводиться оцінювати по навчальній вибірці. В результаті баєсів алгоритм перестає бути оптимальним, адже відновити щільність по вибірці можливо тільки з деякою похибкою. Чим коротше вибірка, тим вище шанси підігнати розподіл під конкретні дані і зіткнутися з ефектом перенавчання. І хоча даний алгоритм є одним з найстаріших, аналіз показує, що він втримує стійкі позиції і сьогодні. Цей підхід лежить в основі й багатьох інших достатньо успішних баєсових методів класифікації. До їх числа відносяться: лінійний дискримінант Фішера, метод парзенового вікна, метод радіальних базисних функцій (RBF) та інші.

1.6.1.1 Наївний баєсів класифікатор

Naive Bayes (наївний баєсів класифікатор) – класифікатор, що є одним з обширної сукупності баєсових методів – алгоритмів заснованих на принципі максимума апостеріорної вірогідності. Для об’єкта, що піддають класифікації обраховують так звані функції правдоподібності кожного з класів, по ним обраховують апостеріорні вірогідності цих класів. Після чого об’єкт буде віднесено до того класу, для якого апостеріорна вірогідність максимальна.

Безпосередньо сам Наївний байесовский алгоритм (далі НБА) - це алгоритм класифікації, заснований на теоремі Байеса з припущенням про незалежність ознак. Іншими словами, НБА передбачає, що наявність якої-небудь ознаки в класі не

пов'язано з наявністю будь-якої іншого ознаки. Наприклад, фрукт може вважатися яблуком, якщо він червоний, круглий і його діаметр становить близько 8 сантиметрів. Навіть якщо ці ознаки залежать один від одного або від інших ознак, в будь-якому випадку вони вносять незалежний внесок у ймовірність того, що цей фрукт є яблуком. У зв'язку з таким припущенням алгоритм називається «наївним».

Теорема Байєса (за роботою [22]) дозволяє розрахувати апостеріорну ймовірність $P\ (y \mid x)$ на основі $P(y)$, $P(x)$ і $P\ (x \mid y)$:

$$P(y \mid x) = \frac{P(x \mid y)\ P(y)}{P(x)}, \tag{1.20}$$

де $P(y|x)$ – posterior probability – апостеріорна ймовірність, що дане значення ознаки x говорить про належність до класу y, саме її нам треба розрахувати;

$P(x|y)$ – likelihood – ймовірність зустріти ознаку x серед всіх ознак що належать класу y (правдоподібність, тобто ймовірність даного значення ознаки при даному класі);

$P(y)$ – class prior probability – безумовна (апріорна) ймовірність зустріти документ класу y в корпусі документів;

$P(x)$ – predictor prior probability – безумовна (апріорна) ймовірність даного значення ознаки.

Простіше для розуміння буде переписати теорему в інших позначеннях:

$$P(\theta|D) = \frac{P(\theta)P(D|\theta)}{P(D)} \tag{1.21}$$

Якщо співвідносити це з типовим завданням машинного навчання, то тут D - це дані, то, що ми знаємо, а θ – це параметри моделі, які ми хочемо навчити. Наприклад, в моделі SVD [23] дані – це ті рейтинги, які ставили користувачі

продуктам, а параметри моделі – фактори, які ми навчаємо для користувачів і продуктів.

Кожна з ймовірностей теж має свій сенс. Так, $P(\theta|D)$ – це те, що ми хочемо знайти, розподіл ймовірностей параметрів моделі після того, як ми прийняли до уваги дані. Це називається апостеріорною ймовірністю (posterior probability). Цю ймовірність, як правило, безпосередньо не знайти, і тут як раз і потрібна теорема Байеса.

Значення $P(D|\theta)$ – це так звана правдоподібність (likelihood), ймовірність даних за умови зафіксованих параметрів моделі. Знайти її зазвичай легко, власне, конструкція моделі зазвичай в тому і полягає, щоб задати функцію правдоподібності. Апріорна ймовірність $P(\theta)$ (prior probability) – є математичною формалізацією нашої інтуїції про предмет, формалізацією того, що ми знали раніше, ще до всіляких експериментів.

Сьогодні баєсів підхід вважають скоріше як розгляд вірогідностей з позиції «ступеня довіри». Для кращого розуміння можна скористатись прикладом використання даного класифікатора при категоризації текстів. Так, якщо необхідно сортувати потік новин по темам на основі вже існуючої бази даних з темами, ми будемо використовувати так звану «bag-of-words» модель [24]: представляти документ (мульти)множиною слів, які в ньому містяться. В результаті кожен тестовий приклад x приймає значення із множини категорій V та описується атрибутами $\langle a_1, a_2, \dots, a_n \rangle$. Нам же необхідно найти найбільш вірогідне значення даного атрибута:

$$v_{map} = \arg\max_{v \in V} P(x = v|a_1, a_2, \dots, a_n) \qquad (1.22)$$

Сама теорема Баєса тут допомагає нам поміняти місцями причину і наслідок. Знаючи з якою вірогідністю причина приводить до деякої події, вона дозволяє розрахувати вірогідність того, що саме ця причина призвела до спостережуваної

події. Ціль класифікації полягає в тому, щоб зрозуміти до якої категорії належить дана новина, тому нам потрібна не сама новина, а найбільш вірогідна категорія. Як було вже зазначено, для її визначення Баєсовий класифікатор використовує оцінку апостеріорного максимума. За теоремою Баєса :

$$v_{map} = \arg\max_{v \in V} \frac{(a_1, a_2, \dots, a_n | x = v) P(x = v)}{P(a_1, a_2, \dots, a_n)} = \\ = \arg\max_{v \in V} P(a_1, a_2, \dots, a_n | x = v) P(x = v) \quad (1.23)$$

В формулі 1.23 в першій частині знаменник (вірогідність події) являється константою та ніяк не впливає на ранжування класів. Оцінити $P(x = v)$ достатньо легко – можна оцінювати частоту зустрічальності. Проте оцінити різні $P(a_1, a_2, \dots, a_n | x = v)$ не вийде, адже їх дуже багато. Тут же $P(a_1, a_2, \dots, a_n | x = v)$ – вірогідність в точності такого набора слів в повідомленнях на різні теми. Стає очевидно, що такої статистики взяти ніде.

Для того, щоб з цим справитись, наївний баєсів класиікатор передбачає умовну незалежність атрибутів при умові даного значення цільової функції:

$$P(a_1, a_2, \dots, a_n | x = v) p(x = v) = \\ = P(a_1 | x = v) P(a_2 | x = v) P(a_n | x = v) \quad (1.24)$$

Тепер навчити окремі $P(a_i | x = v)$ набагато простіше: достатньо підрахувати статистику зустрічальності слів у категоріях.

Слід зауважити, що наївний баєсовий класифікатор робить досить сильне припущення: в класифікації текстів ми припускаємо, що різні слова в тексті на одну і ту ж тему з'являються незалежно один від одного. Таке припущення в хоч і виглядає дещо нелогічним, проте це на практиці дає гарні результати. Насправді наївний баєсовий класифікатор набагато краще, ніж здається. Його оцінки

ймовірностей оптимальні, звичайно, тільки в разі справжньої незалежності, але сам класифікатор оптимальний в куди більш широкому класі задач, і ось чому.

По-перше, атрибути, звичайно, залежні, але їх залежність однакова для різних класів і «взаємно скорочується» при оцінці ймовірностей. Граматичні та семантичні залежності між словами одні й ті ж самі, що в тексті на вільну тему, що в тексті про Баєсове навчання. По-друге, для оцінки ймовірностей наївний Байес дуже поганий, але як класифікатор набагато кращий (звичайно, якщо навіть насправді $P(x = v_0|D) = 0.51; P(x = v_1|D) = 0.49$, наївний Байес видасть $P(x = v_0|D) = 0.99; P(x = v_1|D) = 0.01$, але класифікація при цьому буде частіше правильна).

1.6.3 Метричні методи

Дана група алгоритмів класифікації, заснована на обчисленні оцінок подібності між об'єктами. Найпростішим метричним класифікатором є метод найближчих сусідів, в якому об’єкт, що класифікується буде віднесеним до того класу, якому належить більшість схожих з ним об'єктів.

Для формалізації поняття подібності вводиться функція відстані між об'єктами $\rho(x, x')$. Як правило, жорстокої вимоги, щоб ця функція була метрикою не пред'являється; зокрема, нерівність трикутника цілком може і порушуватись.

1.6.3.1 Метод *k*-найближчих сусідів

Метод *k*-найближчих сусідів (*k*-nearest neighbours, k-NN)- найпростіший метричний класифікатор, заснований на оцінюванні подібності об'єктів. Класифікується об'єкт відноситься до того класу, якому належать найближчі до нього об'єкти навчальної вибірки.

Математично його можна сформулювати наступним чином. Нехай задана навчальна вибірка пар «об'єкт-відповідь» $X^m = \{(x_1, y_1), \dots, (x_m, y_m)\}$. Нехай на множина об'єктів задана функція відстані $\rho(x, x')$. Ця функція повинна бути досить

адекватною моделлю подібності об'єктів. Чим більше значення цієї функції, тим менш схожими є два об'єкти (x, x').

Для довільного об'єкта u розташуємо об'єкти навчальної вибірки x_i в порядку зростання відстаней до u:

$$\rho(u, x_{1;\,u}) \leq \rho(u, x_{2;\,u}) \leq \cdots \leq \rho(u, x_{m;\,u}), \tag{1.25}$$

де через $x_{i;u}$ позначається той об'єкт навчальної вибірки, який є i-м сусідом об'єкта u.

Аналогічне позначення введемо і для відповіді на i-му сусіді: $y_{i;u}$. Таким чином, довільний об'єкт u породжує свою перенумерацію вибірки. У найбільш загальному вигляді алгоритм найближчих сусідів є:

$$a(u) = \arg\max_{y \in Y} \sum_{i=1}^{m} [y(x_{i;u}) = y] w(i, u), \tag{1.26}$$

де $w(i, u)$ - задана вагова функція, яка оцінює ступінь важливості i-го сусіда для класифікації об'єкта u. Природно вважати, що ця функція невід'ємна та не зростає по i.

По-різному задаючи вагову функцію, можна отримувати різні варіанти методу найближчих сусідів.

- $w(i, u) = [i = 1]$ – найпростіший метод найближчого сусіда;
- $w(i, u) = [i \leq k]$ – метод k-найближчих сусідів;
- $w(i, u) = [i \leq k] q^i$ – метод k-експоненціально зважених найближчих сусідів, де передбачається $q < 1$;
- $w(i, u) = K\left(\frac{\rho(u, x_{i;u})}{h}\right)$ – метод парзенового вікна фіксованої ширини h;

- $w(i,u) = K\left(\frac{\rho(u,x_{i;u})}{\rho(u,x_{k+1;u})}\right)$ – метод парзенового вікна змінної ширини.

У вище наведених прикладах $K(r)$ – задана невід'ємна монотонно незростаюча функція на $[0,+\infty)$, ядро згладжування.

На рисунку 1.7 схематично зображена робота методу. Тестовий зразок (зелене коло) повинен бути класифікований як синій квадрат (клас 1) або як червоний трикутник (клас 2). Якщо $k = 3$, то класифікується як 2-й клас, тому що всередині меншого кола 2 трикутника і тільки 1 квадрат. Якщо $k = 5$, то він буде класифікований як перший клас (3 квадрата проти 2-ох трикутників всередині більшого кола).

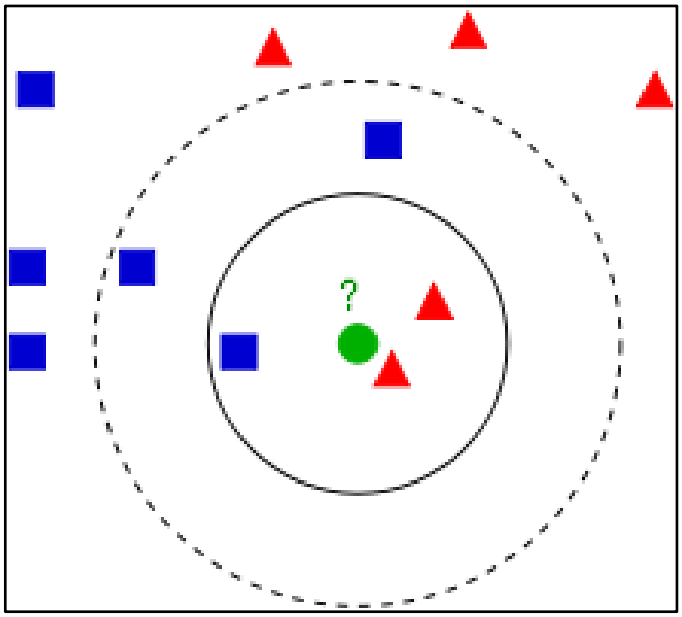

Рисунок 1.7 – Приклад класифікації k-найближчих сусідів

Основні проблеми з якими можна зіткнутись під час використання:

- Вибір числа сусідів k. При $k = 1$ алгоритм найближчого сусіда нестійкий до шумових викидів: він дає помилкові класифікації не тільки на самих об'єктах–викидах, а й на найближчих до них об'єктах інших класів. При $k = m$, навпаки, алгоритм надмірно стійкий і вироджується в константу. Таким чином, крайні значення k

небажані. На практиці оптимальне значення параметра k визначають за критерієм змінного контролю, найчастіше – методом виключення об'єктів по одному (leave–one–out cross–validation) [25].

- Відсіювання шуму (викидів). Зазвичай об'єкти навчання не є рівноцінними. Серед них можуть знаходитися типові представники класів – еталони. Якщо об'єкт для класифікації близький до ідеалу, то, швидше за все, він належить до того ж класу. Ще одна категорія об'єктів – неінформативні або периферійні. Вони щільно оточені іншими об'єктами того ж класу. Якщо їх видалити з вибірки, це практично не позначиться на якості класифікації. Нарешті, до вибірки може потрапити кілька шумових викидів – об'єктів, що знаходяться «в гущі» чужого класу. Як правило, їх видалення тільки покращує якість класифікації. Відкидання з вибірки шумових і неінформативних об'єктів дає кілька переваг одночасно: підвищується якість класифікації, скорочується обсяг збережених даних і зменшується час класифікації, що витрачається на пошук найближчих еталонів.

При використанні даного підходу існує проблема вибору метрики. Це найбільш складна з усіх проблем. У практичних завданнях класифікації рідко зустрічаються такі «ідеальні випадки», коли заздалегідь відома хороша функція відстані $\rho(x, x')$. Якщо об'єкти описуються числовими векторами, часто беруть евклідову метрику. Цей вибір, як правило, нічим не обгрунтований – просто це перше, що спадає на думку. При цьому необхідно пам'ятати, що всі ознаки мають бути виміряні «в одному масштабі», а найкращий випадок коли вони будуть віднормовані. В іншому випадку ознака з найбільшими числовими значеннями буде домінувати в метриці, тоді як інші ознаки, фактично, враховуватися не будуть. Однак і нормування є вельми сумнівною евристикою, так як залишається питання: «Невже всі ознаки однаково значущі і повинні враховуватися приблизно з однаковою вагою?».

Якщо ознак занадто багато, а відстань обчислюється як сума відхилень за окремими ознаками, то виникає проблема «прокляття розмірності». Суми великого числа відхилень з великою ймовірністю мають дуже близькі значення (відповідно до закону великих чисел). Виходить, що в просторі високої розмірності всі об'єкти приблизно однаково далекі один від одного; вибір k-найближчих сусідів стає практично довільним. Проблема вирішується шляхом відбору відносно невеликого числа інформативних ознак (features selection). В алгоритмах обчислення оцінок будується множина різних наборів ознак (так званих «опорних множин») [26], для кожного будується своя функція схожості, потім по всім функціям схожості проводиться голосування.

1.6.4 Методи на основі правил ухвалення рішень

Дану групу методів також можна віднести до більш загальної категорії «Логічних алгоритмів класифікації» (за К.В. Воронцовим) [27], що застосовують в класифікаторах, що побудовані на основі вирішаючих правил, коли простір даних моделюється набором правил, на лівій стороні якого знаходиться умова на набір признаків, а на правій – мітка класу.

Набір же самих правил генерується з навчальної вибірки. Для даного тестового об’єкта ми отримуємо набір правил, для котрих він задовольняє їх умові на лівій стороні. Після чого визначається мітка класу як функція від отриманих міток класу правил. В більш загальному вигляді, ліва сторона правила являється логічною умовою, що виражена зазвичай в диз’юнктивній нормальній формі (ДНФ). Тим не менш, в більшості випадків, умова на лівій стороні значно простіша і представляє собою набір термів, всі з яких повинні знаходитись у вхідному тексті, для задоволення умови.

Одним із основних принципів при побудові набору правил являється те, що простір відповідей повинен бути покритим хоча б одним правилом. В більшості випадків, це досягається побудовою різноманітних наборів правил для кожного класу та одного правила «за змовчуванням», яке покриватиме всі інші екземпляри.

1.6.4.1 Дерево ухвалення рішень

Дерево ухвалення рішень (Decision tree) – розв’язання задачі навчання з учителем, засноване на тому, як розв’язує завдання прогнозування людина. У загальному випадку - це k-те дерево з вирішальними правилами в нелистових вершинах (вузлах) і деякому висновку про цільову функцію в листових вершинах (прогнозом). Вирішальне правило - деяка функція від об'єкта, що дозволяє визначити, в яку з дочірніх вершин потрібно помістити даний об'єкт. У листових вершинах можуть перебувати різні об'єкти: клас, який потрібно присвоїти потрапившому туди об'єкту (в задачах класифікації), ймовірності класів (в задачах класифікації), безпосередньо значення цільової функції (задачі регресії).

Деревом називається кінцевий зв’язний граф с множиною вершин V, що не містить циклів та має виділену вершину $v_0 \in V$, до котрої не входить ні одне ребро. Така вершина називається коренем, а вершина, що не має вихідних ребер називається термінальною або листом. Решта вершин звуться внутрішніми.

Найчастіше на практиці використовуються двійкові вирішальні дерева, що також звуться бінарними. Дерево називається бінарним, якщо з будь-якої внутрішньої вершини виходить рівно два ребра. Вихідні ребра зв’язують кожну внутрішню вершину v з лівою дочірньою вершиною L_v та з правою дочірньою вершиною R_v

Бінарне дерево ухвалення рішень – це алгоритм класифікації, що задається бінарним деревом, в якому кожній внутрішній вершині $v \in V$ приписаний предикат $\beta_v: X \to \{0,1\}$, кожній термінальній вершині $v \in V$ приписане ім’я класу $c_v \in Y$. При класифікації об’єкта $x \in X$ він проходить по дереву шлях від кореня до деякого листа, згідно алгоритму:

1. $v := v_0$;
2. поки вершина v внутрішня
3. якщо $\beta_v(x) = 1$ то
4. $v := R_v$; (перехід вправо)

5. інакше
6. $v := L_v$; (перехід вліво)
7. повернути c_v.

На рисунку 1.8 зображене дерево в загальному випадку.

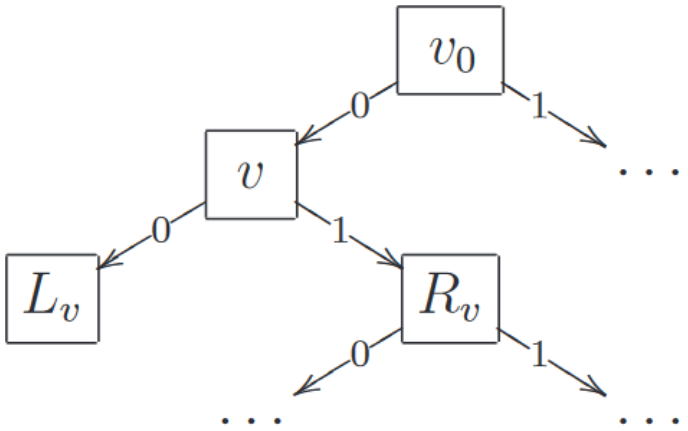

Рисунок 1.8 – Класифікація об'єкту $x \in X$ бінарним деревом ухвалення рішень

Об'єкт x входить до вершини v тоді і тільки тоді, коли виконується кон'юнкція $K_v(x)$, що складається із усіх предикатів, прописаних внутрішніми вершинами дерево на шляху від кореня v_0 до вершини v. Нехай T – множина усіх термінальних вершин дерева. Множина об'єктів $\Omega_v = \{x \in X: K_v(x) = 1\}$, що виділені термінальними кон'юнкціями $v \in T$, попарно не пересікаються, а їх об'єднання співпадає з усім простором X (це доводиться індукцією по числу вершин дерева). Звідси слідує, що рішаюче дерево ніколи не відмовляється від класифікації, на відміну від вирішаючого списку [27]. Звідси також слідує, що алгоритм класифікації $a: X \to Y$, що реалізований бінарним рішаючим деревом, можна представити у вигляді простого голосування кон'юнкцій:

$$a(x) = \arg\max_{y \in Y} \sum_{v \in T} [c_v = y] K_v(x) \qquad (1.27)$$

При чому для любого $x \in X$ один і тільки один доданок в усіх цих сумах рівний одиниці. Замість додавання можна було б використовувати і диз'юнкцію.

Як зазначає К. В. Воронцов, вимога максимізації інформативності кон'юнкції $K_v(x)$ означає, що кожна з них повинна виділяти як можна більше навчальних об'єктів, допускаючи при цьому як можна менше похибок. Для підвищення узагальнюючої здатності вирішального дерева кількість листя повинно бути як можна меншим і вони повинні покривати підвибірки приблизно однакової потужності $\Omega_v \bigcap X^l$.

1.6.5 Штучні нейронні мережі

Мережі з прямим зв'язком є універсальним засобом апроксимації функцій, що дозволяє їх використовувати в рішенні задач класифікації. Як правило, нейронні мережі виявляються найбільш ефективним способом класифікації, тому що фактично генерують велике число регресійних моделей (які використовуються в рішенні задач класифікації статистичними методами).

На жаль, в застосуванні нейронних мереж в практичних завданнях виникає ряд проблем. По-перше, заздалегідь не відомо, якої складності (розміру) може знадобитися мережа для досить точної реалізації відображення. Ця складність може виявитися надмірно високою, що потребує складної архітектури мереж. Так Мінський в своїй роботі "Персептрони" [28] довів, що найпростіші одношарові нейронні мережі здатні вирішувати тільки лінійно роздільні завдання.

Це обмеження можна подолати при використанні багатошарових нейронних мереж. У загальному вигляді можна сказати, що в мережі з одним прихованим шаром вектор, відповідний вхідному зразку, перетворюється прихованим шаром в якийсь новий простір, який може мати іншу розмірність, а потім гіперплощини,

відповідні нейронам вихідного шару, поділяють його на класи. Таким чином мережа розпізнає не тільки характеристики вихідних даних, але і "характеристики характеристик", сформовані прихованим шаром.

Завдання класифікації при наявності двох класів може бути вирішена на мережі з одним нейроном у вихідному шарі, який може приймати одне з двох значень 0 або 1, залежно від того, до якого класу належить зразок. При наявності декількох класів виникає проблема, пов'язана з представленням цих даних для виходу мережі. Найбільш простим способом представлення вихідних даних в такому випадку є вектор, компоненти якого відповідають різним номерам класів.

При цьому і-а компонента вектору відповідає і-му класу. Всі інші компоненти при цьому встановлюються в 0. Тоді, наприклад, другому класу буде відповідати «1» на 2 виході мережі і 0 на інших. При інтерпретації результату зазвичай вважається, що номер класу визначається номером виходу мережі, на якому з'явилося максимальне значення. Наприклад, якщо в мережі з трьома виходами ми маємо вектор вихідних значень (0.2, 0.6, 0.4), то ми бачимо, що максимальне значення має друга компонента вектору, значить клас, до якого належить цей приклад – клас 2.

При такому способі кодування іноді вводиться також поняття впевненості мережі в тому, що приклад відноситься до цього класу. Найбільш простий спосіб визначення впевненості полягає у визначенні різниці між максимальним значенням виходу і значенням іншого виходу, яке є найближчим до максимального. Наприклад, для розглянутого вище прикладу впевненість мережі в тому, що приклад відноситься до другого класу, визначиться як різниця між другою і третьою компонентою вектору і дорівнює 0.6-0.4 = 0.2. Відповідно чим вище впевненість, тим більша ймовірність того, що мережа дала правильну відповідь. Цей метод кодування є найпростішим, але не завжди найоптимальнішим способом представлення даних.

1.6.6 Результат аналізу наявних методів

В даному підрозділі наведені основні аспекти та сторони описаних вище, в даному розділі, методів, що являються широковживаними в задачах класифікації.

Позитивні і негативні сторони методу опорних векторів

Позитивні сторони:

- Це найбільш швидкий метод знаходження вирішальних функцій;
- Один з найкращих методів для бінарної класифікації.
- Метод зводиться до вирішення задачі квадратичного програмування в опуклій області, яка завжди має єдине рішення;
- Метод знаходить розділяючу смугу максимальної ширини, що дозволяє в подальшому здійснювати більш впевнену та якісну класифікацію;

Негативні сторони:

- Метод чутливий до шумів і стандартизації даних;
- Не існує загального підходу до автоматичного вибору ядра в разі лінійної нероздільності класів.

Позитивні і негативні сторони логістичної регресії

Позитивні сторони:

- Швидкість і простота отримання моделі.
- Інтепретованість моделі. Лінійна модель є прозорою і зрозумілою для аналітика. За отриманими коефіцієнтами регресії можна судити про те, як той чи інший фактор впливає на результат, зробити на цій основі додаткові корисні висновки.
- Широка застосовність. Велика кількість реальних процесів в економіці та бізнесі можна з достатньою точністю описати лінійними моделями.

- Вивченість даного підходу. Для лінійної регресії відомі типові проблеми і їх рішення, розроблені та реалізовані тести оцінки статичної значущості одержуваних моделей.

Негативні сторони:

- Низька стійкість до помилок;
- Залежність від набору даних;
- Градієнтний метод навчання логістичної регресії наслідує всі недоліки методу стохастичного градієнта. Практична реалізація повинна передбачати стандартизацію даних, регуляризацію (скорочення ваг), відбір признаків та інші евристики для покращення збіжності.

Позитивні і негативні сторони наївного баєсівського алгоритму

Позитивні сторони:

- Перевагою наївного баєсівського класифікатора є мала кількість даних для навчання, необхідних для оцінки параметрів, що необхідні для класифікації;
- Моделі на основі НБА досить прості і вкрай корисні при роботі з дуже великими наборами даних. При своїй простоті НБА здатний перевершити навіть деякі складні алгоритми класифікації.
- Класифікація, в тому числі багато класова, виконується легко і швидко.
- Коли допущення про незалежність виконується, НБА перевершує інші алгоритми, такі як логістична регресія (logistic regression), і при цьому вимагає менший обсяг навчальних даних.
- НБА краще працює з категорійними ознаками, ніж з безперервними. Для безперервних ознак передбачається нормальний розподіл, що є досить сильним допущенням.

Негативні сторони:

- Якщо в тестовому наборі даних є певне значення категорійної ознаки, яке не зустрічалося в навчальному наборі даних, тоді модель присвоїть нульову ймовірність цього значення і не зможе зробити прогноз. Це явище відоме під назвою «нульова частота» (zero frequency). Дану проблему можна вирішити за допомогою згладжування. Одним з найпростіших методів є згладжування по Лапласу (Laplace smoothing) [29].
- Хоча НБА є хорошим класифікатором, значення прогнозованих ймовірностей не завжди є достатньо точними. Тому не слід занадто покладатися на результати.
- Ще одним обмеженням НБА є припущення про незалежність ознак. В реальності набори повністю незалежних ознак зустрічаються вкрай рідко.

Позитивні і негативні сторони k-NN

Позитивні сторони:

- Простота реалізації.
- Класифікацію, проведену даним алгоритмом, легко інтерпретувати шляхом пред'явлення користувачеві декількох найближчих об'єктів.
- Метою пошуку є не гарантовано вірне рішення, а найкраще з можливих.

Негативні сторони:

- Не ефективна витрата пам'яті і надмірне ускладнення вирішального правила внаслідок необхідності зберігання навчальної вибірки повністю;
- Пошук найближчого сусіда передбачає порівняння об'єкта класифікації з усіма об'єктами вибірки, що потребує великого числа операцій;

- Існує складність вибору міри "близькості" (метрики). Від неї головним чином залежить обсяг множини записів, які потрібно зберігати в пам'яті для досягнення задовільної класифікації або прогнозу. Також існує висока залежність результатів класифікації від обраної метрики.

Позитивні і негативні сторони дерева ухвалення рішень

- Простий в розумінні та інтерпретації.
- Люди здатні інтерпретувати результати моделі дерева ухвалення рішень після короткого пояснення
- Не потребує підготовки даних. Інші техніки вимагають нормалізації даних, додавання фіктивних змінних, а також видалення пропущених даних.
- Здатний працювати як з категоріальними, так і з інтервальними змінними. Інші методи працюють лише з одним типом.
- Використовує модель «білого ящика». Якщо певна ситуація спостерігається в моделі, то її можна пояснити за допомогою булевої логіки. Прикладом «чорного ящика» може бути штучна нейронна мережа, так як результати даної моделі важко піддаються поясненню.
- Добре працює навіть в тому випадку, якщо були порушені початкові припущення, включені в модель.
- Дозволяє працювати з великим об'ємом інформації без спеціальних підготовчих процедур.

Негативні сторони:

- Модель на базі даного методу занадто схильна до перенавчання;
- Узагальнююча властивість дуже низька;
- Висока чутливість до шумів, до критерія інформативності;
- Висока чутливість до складу вибірки. Зміна даних в 1-2 об’єктах може радикально змінити структуру дерева;

- Локально оптимальний вибір предиката не є оптимальний в глобальному розумінні. В разі хибного вибору, алгоритм не здатний повернутись на рівень вгору і замінити невдалий предикат.

1.6.7 Аналіз ринку існуючих рішень

Аналіз ринку показав, що фактично потенційні конкуренти відсутні. Існують величезні системи до котрих можна підключити відповідний функціонал для визначення образ, але з певними обмеженнями.

Так, наприклад є програма Proxomitron - універсальний фільтр для інтернет-сторінок, технічно реалізований як проксі-сервер [30]. Основне його призначення — знищення всіляких видів інтернет-реклами (зокрема банерів) і блокування випливаючих вікон. Додаткові можливості програми дозволяють змінити на сторінці майже все «до останнього тегу», що робить його універсальним інструментом для керування вмістом сторінок. Наприклад, можна додати відсутні можливості, вбудовуючи скрипти, або обійти різні обмеження та захист на веб-сайтах. Аналогічним варіантом є программа HandyCache (HC) [31].

Аналіз показує, що можливість реалізації просто фільтру контенту в обох додатках (наприклад, просто видалення образливих записів і тд) можливе тільки за допомогою скриптів, які базуються на пошуку слів із списку, регулярних виразів чи певним мовних правил. Такі скрипти доводиться шукати власноруч, тож вони не являються частиною продукту чи окремим модулем від розробників.

1.7 Висновки за розділом

В даному розділі коротко висвітлено основи теорії класифікації текстів. На основі аналізу існуючих методи машинного було складено порівняльну характеристику, що дозволило визначити найбільш загальновживані. Також було визначено, що вхідний текст потрібно піддавати певній обробці та представляти у вигляді, придатному до опрацювання математичними засобами. На практиці

зазвичичай його представляють у векторному вигляді. В ході досліджень наявних моделей машинного навчання було визначено, що найпопулярнішими є наступні: Naive Bayes (наївний баєсів класифікатор), SVM (метод опорних векторів), та Logistic Regression (логістична регресія) для класифікації текстів.

Короткий аналіз дає розуміння того, що стандартні підходи для класифікації текстів дають гарні результати при наявності великої кількості слів. Проте при класифікації текстів з малою кількістю слів корисної інформації в них значно менше, а значить і кількість ознак, що можна добути з них, також мала. Це в свою чергу негативно впливає на кінцевий результат.

2 РОЗРОБКА МАТЕМАТИЧНОЇ МОДЕЛІ

Даний розділ присвячений розробці математичної моделі в області розпізнавання образ у коротких текстах. Далі буде висвітлене дослідження та аналіз підходу для класифікації. Навдені висновки та запропоновані рішення базуються на результаті вивчення предметної області та аналізу існуючих засобів.

Огляд ринку показав, що на сьогоднішній день ефективних засобів, як безкоштовних, так і комерційних немає. Тому порівняння з існуючими системами здійснити неможливо.

В той же час з того, що вдалось з'ясувати, аналізуючи з наукової точки зору, наявні досить розмиті підходи, для рішення такого роду задачі. Наприклад, існують системи, що призвані класифікувати тексти за темами. Проте в такому випадку, хоча і задача виглядає складнішою, адже відноситься не до бінарної класифікації, а до мультикласової, такі тексти мають значно більші обсяги та наповнені багатьма словами, що несуть змістове навантаження. Основною складністю в рішенні даної теми являється малий обсяг текстів.

2.1 Вибір базового методу машинного навчання

В ході досліджень наявних моделей машинного навчання було визначено що найпопулярнішими являються: Naive Bayes (наївний баєсів класифікатор), SVM (метод опорних векторів), та Logistic Regression (логістична регресія) для класифікації текстів.

Щоб визначити який метод машинного навчання взяти за основу для розв’язання задачі класифікації коротких текстів за наявністю в них образ, необхідно реалізувати кожен з трьох підходів та оцінити показники базових моделей. Так як на даному етапі необхідно лише визначити найкращий метод, вхідний текст для навчання буде піддано базовій обробці.

Для розв’язання поставленої задачі було застосовано кожний з підходів з словесними 1-грамами (unigram), утвореними з вхідних текстів. До тексту було

вирішено застосувати прості функції видалення знаків пунктуації та приведення літер до нижнього регістру.

В якості метрики для оцінки якості було обрано перехресну валідацію, що дозволяє оцінити точність класифікації на даних, які ще не були подані до класифікатора. Це дозволяє отримати більш реальні показники якості класифікації. Для порівняння також було виміряно точність навчання. Результати зображено на рисунку 2.1.

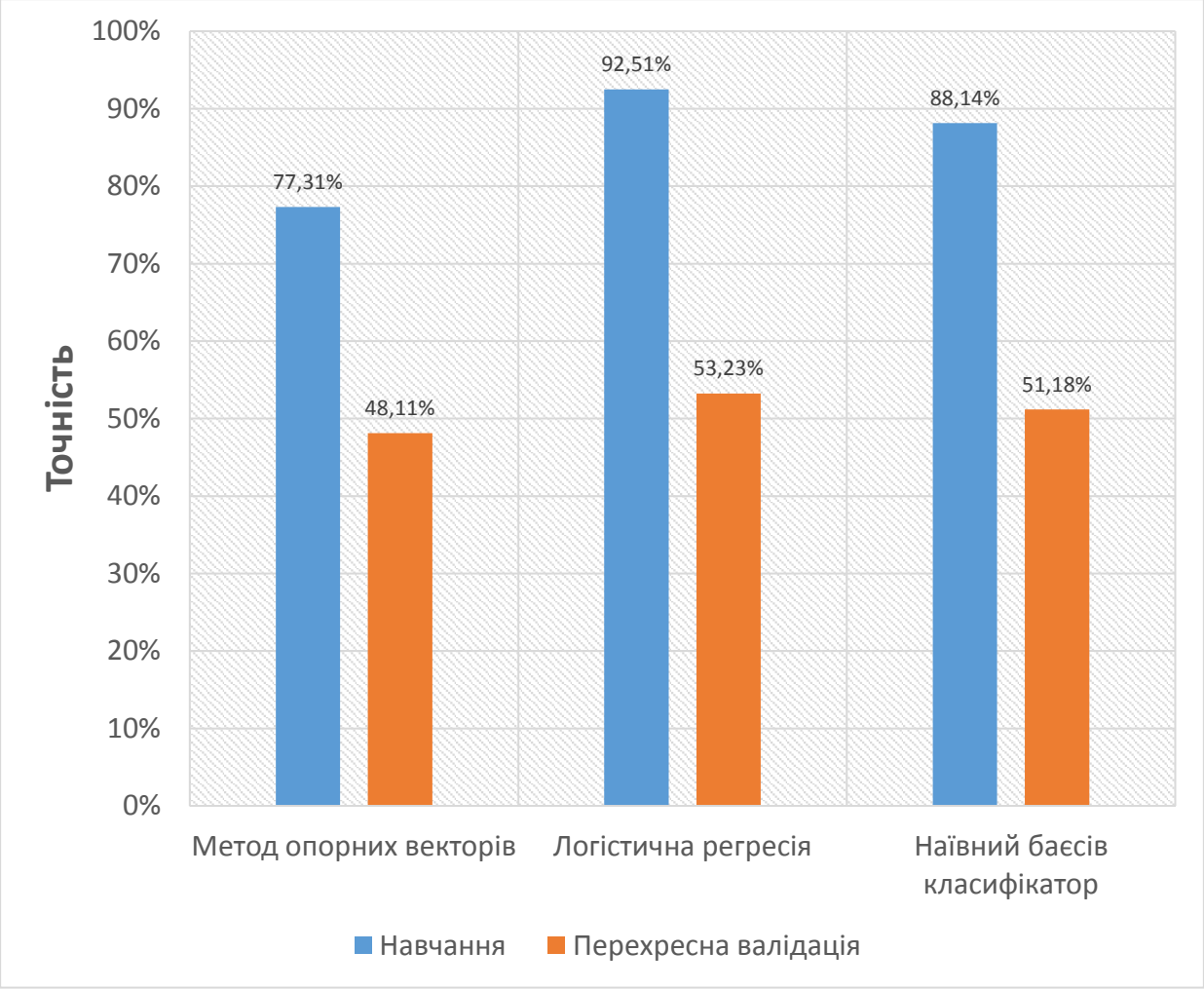

Рисунок 2.1 – Порівняння базових моделей

Порівнюючи показники точності на навчальному наборі та точності виміряної за допомогою перехресної валідації стає очевидним, що в кожному з трьох випадків базові класифікатори перенавчаються. Проте між собою вони не відрізняються дуже суттєво.

Дивлячись на показники для перехресної валідації, видно, що модель з логістичною регресією (Logistic Regression) справляється краще ніж дві інші моделі. Тож було вирішено використати даний підхід як базовий для системи класифікації коротких текстів.

2.2 Пошук вхідних даних

Однією з основних складностей в рішенні задач класифікації, що базуються на методах навчання по прецедентах являється пошук необхідних даних, на яких ми зможемо навчати нашу систему.

Часто на пошук таких витрачається досить багато часу. Даних для навчання в нашому випадку достатньо багато на просторах інтернету. Достатньо зайти на перші сторінки пошукових запитів на тему форумів, соціальних мереж і швидко можна знайти перші приклади. Проте необхідно їх вибрати та зробити необхідну роботу для підготовки та позначення знайдених коментарів на приналежність до категорії образливих чи ні.

Пошук готової зібраної інформації закінчився успіхом. У вигляді даних для навчання майбутньої системи та тестування було обрано набір з сайту Kaggle [32], який є найпопулярнішим ресурсом та площадкою, де проводяться змагання з машинного навчання. Набір даних містить 6594 прикладів в купі з двох файлів (кінцевий класифікатор буде навчатись на такій кількості), де перший файл містить 3948 записів, решта – інший, що був призначений для визначення переможця (пізніше для нього стали доступні й правильні відповіді). Кожний приклад складається з реального тексту, взятого з просторів інтернету та позначки, що свідчить про його забарвленість (1 – образливий, 0 – не образливий). Приблизне співвідношення класифікованих текстів 3/8 відповідно.

2.3 Базові математичні моделі

Постановка задачі звучить наступним чином: існує множина об'єктів X, існує також множина класів Y, і кожен об'єкт належить якомусь з класів в Y. Існує скінченна множина функцій f_i: $X \to \Re$, що називають ознаками об'єкта. Кожному з об'єктів можна поставити у відповідність певний вектор ознак:

$$g: X \to R^n,\ g(x) = (\xi_1, \xi_2, \dots, \xi_n) = (f_1(x), f_2(x), \dots, f_n(x)), \qquad (2.1)$$

де ξ_i – значення i-ої ознаки об'єкта x. Надалі замість конкретно об'єкта (тексту) будемо розглядати відповідний йому вектор ознак.

В даній магістерській дисертації, результати якої будуть наведені далі, була висунута гіпотеза, що використання буквених n-грам як характеристик тексту, в купі з методами машинного навчання може дати гарний кінцевий результат при класифікації тексту на наявність образливих оборотів, так як це може бути в коментарях до записів з соціальних мереж.

До такого висновку наштовхує те, що одним із найчастіше вживаних методів для класифікації текстів являється використання звичайних n-грам в поєднанні з будь-яким методом машинного навчання. Зазвичай в ролі останнього використовують метод опорних векторів SVM (*support vector machine*) чи логістичну регресію (Logistic Regression).

Проте, в силу властивостей та специфіки соціальних мереж, вхідний текст має переважно невелику кількість слів. Це негативно впливає на якість розпізнавання та віднесення до певного класу при використанні n-грам. Тексту ж буде відповідати велика кількість буквених n-грам. Останні успішно використовуються для класифікації текстів по їх авторству [12], коли їх поєднують з використанням статичних критеріїв.

Далі було розглянуті наступні варіанти:

- Кожному тексту ставиться у відповідність чисельний вектор характеристик, що буде складатись з частоти зустрічань в ньому словесних n-грам при $n \leqslant 3$. Або ж індикаторів наявності n-грам при $n > 3$.

- Для уникнення великої кількості характеристик є варіант використання відстані між розподіленнями. Для цього необхідно для кожного заданого n розрахувати розподілення буквених n-грам на даному тексті. Після цього розраховується відстань від даного розподілення до розподілення буквених n-грам на множині текстів з позитивним забарвленням і відсутністю в ньому образ та з фразами, що їх мають. В якості відстані можна взяті наступні функції:

$$L_1(\mathrm{p},\mathrm{q}) = \sum_i |p_i - q_i|\,; \tag{2.2}$$

$$L_2(\mathrm{p},\mathrm{q}) = \sum_i (p_i - q_i)^2\,; \tag{2.3}$$

$$D_{KL}(\mathrm{p},\mathrm{q}) = \sum_i \ln(\frac{p_i}{q_i})p_i. \tag{2.4}$$

Тут p, q – дискретний розподіл з ймовірністю i-го значення p_i і q_i відповідно, D_{KL} – відстань Кульбака-Лейблера між двома розподілами, яка використовувалась в тому числі при класифікації текстів [33]. Таким чином, якщо використовувати m відстаней, то даному тексту при даному n у нас буде можливість поставити у відповідність $2m$ відстаней: m відстаней до розподілу n-грам на множині текстів, що не мають образ та m відстаней до множини текстів з їх присутністю.

2.4 Визначення комплектів характеристик тексту

В якості характеристики тектсу часто використовують частоти слів в них. Так, використовують модель «bag-of-words» [24] для представлення текстів. Крім частоти слів іноді враховують частоти пунктуаційних знаків. Останній підхід не дасть ніякого позитивного впливу, адже, виходячи з того, що цільовими текстами являються коментарі – знаки пунктуації зачасту можуть бути відсутні.

Намагаючись проаналізувати можливі ознаки, що можна виділити з тексту, слід сказати про розподіл довжин слів враховуючи середню довжину слів в тексті. Такий підхід запропоновано в роботах [5-7].

Аналогічно до такого підходу можна також запропонувати використовувати ознаки у вигляді середньої довжини речення в словах, середньої довжини слова в символах та середньої довжини речення в символах. Це дозволить збільшити кількість ознак, що можна виділити, та уникнути недонавчання. Воно трапляється тоді, коли статистична модель або алгоритм машинного навчання не можуть схопити тенденцію, що лежить в основі даних. Кажучи другими словами, модель не достатньо пристосовується до даних. Часто воно є результатом занадто простої моделі.

Якщо не вдаватись в складнощі в процесу виділення частин мови, їх розподіли також можна використовувати як додаткову характеристику тектсу. Так, в роботах [1-3] розглядався розподіл частин мови на декількох перших позиціях та в кінці речення.

Додатковою характеристикою в розрізі групи синтаксичних характеристик також беруть розподіл часто вживаних в мові слів. Існують частотні словники, де вказується кількість зустрічань певного слова на мільйон інших в колекції текстів; [8, 9]. Така характеристика повинна дати гарний результат, проте вона накладає певні обмеження на майбутній розвиток, адже у вільному користуванні вдалось знайти словник тільки для англійської мови, а самостійне складання такого можна винести в тему окремої наукової роботи.

Швидкий аналіз вибірки коментарів та порівняння образливих і звичайних дозволяє помітити, що тексти в першій категорії частіше мають слова, що написані заголовними літерами. Можна враховувати наявність таких слів в тексті. Також, часто можна побачити коментарі, в яких існують слова, що оточені астериксами. Таким чином користувачі соціальних мереж й інших веб-ресурсів іноді передають сарказм. Така ознака має місце бути, проте її вплив може негативно сказатись на якості класифікації в залежності від тематики, адже іноді в астерикси заключають і нейтральну інформацію (наприклад, звичайна цитата). В залежності від цільового

джерела текстів, якщо, наприклад, це є соціальна мережа, де користувачі часто додають смайли чи хеш теги то своїх постів, можна враховувати і наявність таких.

Синтаксичні характеристики на цьому не закінчуються, проте, як можна побачити, частина з них уже сильно залежить від тематики ресурсу, з якого беруться вхідні тексти. Тому, якщо говорити про універсальність, то кількість ознак одразу зменшується в вигоду уникнення помилок, завдяки не вірно обраним для конкретної тематики характеристик.

Основною складністю в застосуванні стандартного підходу для класифікації текстів у вигляді використання методу машинного навчання та словесних n-грам, являється мала довжина текстів в словах. В причину цього буде виділено менше ознак, ніж якби це був здоровий за обсягом текст. Для покращення результату використання n-грам, можна запропонувати використовувати словесні n-грами, де спробувати змінити параметр n в межах від 1 до 3. Також можна використовувати їх як окрему характеристику в купі з іншими характеристиками.

Розвиваючи тему n-грам, запропоновано перейти від словесних n-грам до буквених n-грам. Значення параметру n доцільно брати починаючи с 2. Змініючи значення від 2 до 6, необхідно проаналізувати яким чином зміниться якість класифікації. Для розширення кількості характеристик буде доцільно також дослідити поєднання словесних та буквених n-грам, поєднання буквених між собою.

Останнім варіантом запропоновано поєднати іншим чином описані можливі характеристики.

Згідно описаних вище позицій було вирішено скласти 5 комплектів характеристик:

1. Синтаксичні:
    - Довжина речення в словах;
    - Середня довжина слова в символах;
    - Середня довжина виразів (речень) в словах;
    - Середня довжина виразів (речень) в символах;
    - Розподіл кількості слів заданої довжини (від 1 до 20);

- Розподілення частин мови;
- Розподілення частин мови, що стоять на першій та другій позиціях в реченні.

2. Буквені *n*-грами. Їх можна розглядати як набір характеристик при одному *n*, так і союз (поєднання) декількох наборів характеристик при різних *n*. Так, наприклад, доцільним буде розглядати 2-грами та 3-грами разом в силу їх відносно невеликої кількості. В такому випадку їм буде відповідати набір характеристик, що складатиметься із частоти зустрічань буквених 2-3-грам.
3. Словесні (звичайні) *n*-грами. Даний підхід являється найпопулярнішим в класифікації документів. В причину малої кількості слів в коментарях будемо розглядати тільки 1-,2-,3-грами.
4. Поєднання буквених *n*-грам із звичайними *n*-грамами, де кожні *n*-грами будуть являтись окремими характеристиками.
5. Інші комбінації попередніх комплектів характеристик між собою.

До запропонованого набору включено словесні (звичайні) n-грами як в ролі самостійних характеристик, так і в купі з іншими. Незважаючи на те, що 1-грами показали низькі результати при виборі базової моделі, їх все одно було залишено для порівняння. Базова модель показала, що здатна до перенавчання. Очевидно, що й інші запропоновані також можуть перенавчитись, що являється достатньо поширеною проблемою при використанні методів машинного навчання. Тому в комплексі вирішення цієї проблеми для обраної логістичної регресії, словесні n-грами можуть дати задовільні результати, коли вплив перенавчання буде нівельовано. Для подальшого дослідження необхідно визначитись, які метрики якості краще підійдуть для класифікатора на основі запропонованих моделей.

2.5 Вибір метрик для оцінки якості моделі

Для визначення ефективності роботи алгоритму доцільніше буде мати декілька метрик, що відображатимуть якість віднесення висловлювань до однієї з двох категорій на тестовому наборі фраз:

tp_x – кількість об’єктів класу *x*, що віднесені алгоритмом до класу *x*

fp_x – кількість об’єктів не класу *x*, що віднесені алгоритмом до класу *x*;

fn_x – кількість об’єктів класу *x*, що віднесені алгоритмом не до класу *x*;

tn_x – кількість об’єктів не класу *x*, що віднесені алгоритмом не до класу *x*;

S – множина усіх класів, до котрих відносяться об’єкти вибірки.

Для основних метрик якості було запропоновано наступні:

- Precision – частка об’єктів, що класифіковані алгоритмом до конкретного класу, та які дійсно відносяться до нього:

$$P = \frac{tp_x}{tp_x + fp_x} \tag{2.5}$$

- Macro Precision – середнє значення для точності по усім класам:

$$\text{Macro\_P} = \frac{1}{|S|} \sum_{x \in S} \frac{tp_x}{tp_x + fp_x} \tag{2.6}$$

- Recall – частка об’єктів конкретного класу, що вірно класифікована алгоритмом і віднесена до нього:

$$R = \frac{tp_x}{tp_x + fn_x} \tag{2.7}$$

- Macro Recall – середнє значення по повноті серед усіх класів:

$$\text{Macro\_R} = \frac{1}{|S|} \sum_{x \in S} \frac{tp_x}{tp_x + fn_x} \tag{2.8}$$

- F1-measure – середнє гармонічне для метрик Precision та Recall:

$$F1 = \frac{2PR}{P + R} \tag{2.9}$$

- Macro F1-measure – середнє по F1-measure серед усіх класів:

$$\text{Macro\_F1} = \frac{1}{|S|} \sum_{x \in \text{S}} F1_x \tag{2.10}$$

- Accuracy – частка вірно класифікованих об’єктів серед усіх об’єктів по усім класам:

$$\text{Accuracy} == \frac{1}{|S|} \sum_{x \in \text{S}} \frac{tp_x + tn_x}{tp_x + fn_x + tn_x + fp_x} \tag{2.11}$$

- K-fold Cross-validation – k-кратна перехресна перевірка. Дана метрика призначена для оцінювання якості класифікатора, що показує наскільки потенційно класифікатор зможе показувати гарні результати на тестових даних. Для цього набір даних для навчання розбивається на k рівних наборів. З даної колекції одна частина береться для тестування, інші k-1 наборів являтиметься навчальним набором. Далі процес навчання повторюється на інших описаних вище наборах. Кінцеве значення являється узагальненим значення точності на кожному кроці. Перевага такого способу полягає в тому, що всі дані приймають участь як в навчанні, так і в тестуванні, де кожне спостереження використовується тільки

раз. На рисунку 2.2 зображений приклад алгоритму k-fold Cross-validation при k = 4.

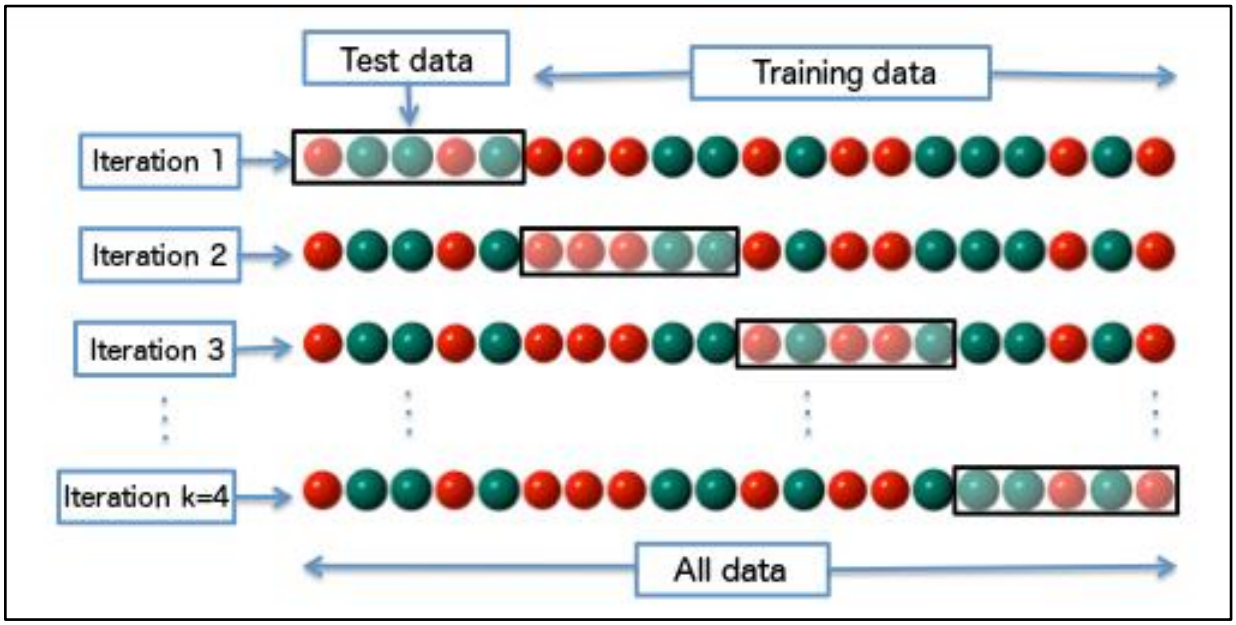

Рисунок 2.2 – Схематичне зображення алгоритму 4-fold Cross-validation

Будемо вважати, що такої кількості метрик, для визначення якості моделі буде достатньо.

2.6 Висновки за розділом

В даному розділі було висвітлено основну проблему класифікації текстів, що мають малий обсяг. На основі порівняння якості класифікації простих моделей реалізованих окремо на одному з трьох методів машинного навчання: Naive Bayes (наївний баєсів класифікатор), SVM (метод опорних векторів) та Logistic Regression (логістична регресія) було визначено Logistic Regression як кращий метод. Його і було взято для подальшого дослідження.

Було складено 5 комплектів характеристик, згідно яких можна добути ознаки з тексту. Основне питання подальшого дослідження – визначення кращого з складених комплектів. Заздалегідь визначити найоптимальніший варіант без практичних випробувань доволі складно. Для підтвердження висунутих

пропозицій їх необхідно перевірити на практиці. Наступний розділ буде присвячений дослідження впливу різних параметрів на результат. В ході дослідження будуть випробувані різні моделі, з яких буде обрана найкраща.

3 ДОСЛІДЖЕННЯ ВПЛИВУ ПАРАМЕТРІВ МАТЕМАТИЧНИХ МОДЕЛЕЙ ДЛЯ БІНАРНОЇ КЛАСИФІКАЦІЇ

Даний розділ присвячено дослідженню та аналізу впливу параметрів системи та запропонованих характеристик на якість класифікації. Для цього було реалізовано кожен з підходів та шляхом порівняння обрано кращий. Інформація про дослідження подана у хронологічній послідовності, а результати відображені у вигляді таблиць та графіків.

3.1 Результати тестування математичних моделей на навчальних даних

Для даної задачі визначення чи являються тексти, що подаються на вхід образливими чи ні, були протестовані набори характеристик, що були описані в підрозділі 2.3.

Для оцінки якості моделі на початковому етапі було застосовано метод k-кратної перехресної перевірки. Дана метрика гарно підходить для оцінки якості класифікатора без використання тестових даних, а тестування відбуватиметься з частин навчальної вибірки, що не відомі класифікатору. Такого роду підхід для оцінки допоможе реально з’ясувати чи перенавчається система чи ні.

Схематично процес перенавчання можна зобразити на площині згідно рисунку 3.1, де можна побачити, що і хоча крива лінія (зелена лінія з багатьма поворотами), що розділяє об’єкти двух класів, краще відповідає зразкам, по котрим відбувалось навчання, проте вона сильно залежить від певних параметрів, і скоріш за все нові дані будуть гірше класифікуватись, ніж якби регуляризована модель (чорна лінія без різких поворотів) була б взята за основу. В першому випадку система являється перенавченою.

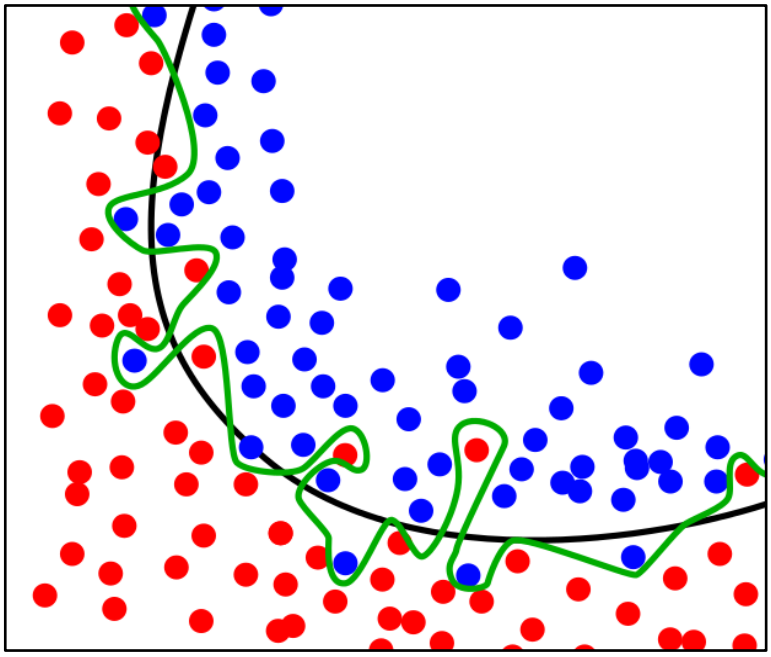

Рисунок 3.1 – Схематичне зображення перенавченої моделі

Для порівняння необхідно навчену систему прогнати на навчальному наборі для визначення метрики *Accuracy*, що вказуватиме на точність класифікації на навчальному наборі. Відповідно до запропонованих наборів для моделей було перевірено кожну з них. Для k-кратної перехресної перевірки було обрані три значення k: k=5; k=10; k=20.

Результати замірів кожного з показнків точності занесено то таблиці 3.1.

Таблиця 3.1 – Значення метрик *CV* та *Accuracy* для різних моделей (початковий тест)

№ гр.	Алгоритм	Метрика			
		5-k fold CV	*10-k fold CV*	*20-k fold CV*	*Accuracy*
1	LR і синт-ні характеристики	0.35	0.38	0.41	0.86
	LR і буквені 2-,3-грамами	0.51	0.60	0.62	0.97
	LR і буквені 4-грами	0.45	0.48	0.59	0.96
	LR і буквені 5-грами	0.47	0.53	0.55	0.94

	LR і буквені 6-грами	0.46	0.51	0.61	0.94
	LR і буквені 2-,3-,4-,5-,6-грами	0.55	0.57	0.7	0.99
	LR і словесні 1-грами	0.51	0.54	0.53	0.93
	LR і словесні 2-грами	0.56	0.55	0.53	0.98
	LR і словесні 3-грами	0.41	0.49	0.58	0.96
	LR і звичайні 1-,2-,3-грами та буквені 2-,3-,4-,5-,6-грами	0.59	0.71	0.75	0.99
	LR і звичайні 1-,2-,3-грами та буквені 4-,5-,6-грами	0.53	0.69	0.70	0.99
	LR і словесні 1-грами з синтакс. х-ми	0.56	0.68	0.81	0.99
	LR і буквені 2-,3-,4-,5-,6-грами з синтакс. х-ми	0.49	0.81	0.85	0.99

Виходячи з результатів, стає очевидно, що система перенавчається та просто запам’ятовує правильні відповіді. На це вказує значення метрики *Accuracy*, в порівнянні із значеннями метрик *CV*. Перша на будь-якому наборі прямує до одиниці, в той час як інші показники суттєво відстають. Це зумовлено тим, що класифікатор гарно опрацьовує уже відомі йому вхідні тексти, проте не має гарних узагальнених знань. Тому на даних, що ще не подавались до системи він часто робить помилки.

3.2 Зменшення впливу перенавчання та результати

Класифікатори, що переначились, не спроможні дати задовільні результати. Узагальнення отриманих знань відбулось погано, що не дозволить проводити якісну класифікацію тестових даних.

Для вирішення цієї проблеми було використано дві речі, що полягають у наступному:

- Змінити метод для визначення ваг. Замість того, щоб використовувати оцінку максимальної правдоподібності, було використано баєсову оцінку максимуму апостеріорної імовірності для визначення ваг (значень шуканих параметрів). Дані підходи тісно пов’язані, але останній застосовує розширену цільову функцію, що включає апріорний розподіл оцінюваної величини. Це сприяє зменшенню норми вектору параметрів, що дозволяє бути менш чутливим до перенавчання.
- Фільтрація вибору ознак. Для цього в кожному випадку експерементальним шляхом було визначено відсоток ознак, що мають найвищий вплив на результати. Беручи до уваги лише частину з усіх добутих ознак, величина вектору ознак буде зменшена, що також позитивно впливає на ситуацію з перенавчанням.

Очевидно, що попередні моделі, як вони є, не являються задовільними, а визначити, яка з них краще по точності класифікації неможливо. Завдяки реалізації вищеописаних функцій вдалось покращити результати для *CV*. В свою чергу кожна з моделей показала суттєво інші показники для метрики *Accuracy*.

Результати впровадження наведено в таблиці 3.2.

Таблиця 3.2 - Значення метрик *CV* та *Accuracy* для різних моделей (зменшення перенавчання)

№ гр.	Алгоритм	Метрика			
		5-k fold CV	*10-k fold CV*	*20-k fold CV*	*Accuracy*
1	LR і синт-ні характеристики	0.66	0.63	0.67	0.82
	LR і буквені 2-,3-грамами	0.52	0.46	0.53	0.74
	LR і буквені 4-грами	0.50	0.48	0.51	0.76

	LR і буквені 5-грами	0.44	0.41	0.44	0.79
	LR і буквені 6-грами	0.54	0.45	0.52	0.81
	LR і буквені 2-,3-,4-,5-,6-грами	0.47	0.48	0.56	0.8
	LR і словесні 1-грами	0.32	0.44	0.41	0.75
	LR і словесні 2-грами	0.41	0.46	0.52	0.88
	LR і словесні 3-грами	0.43	0.49	0.50	0.74
	LR і звичайні 1-,2-,3-грами та буквені 2-,3-,4-,5-,6-грами	0.33	0.54	0.59	0.88
	LR і звичайні 1-,2-,3-грами та буквені 4-,5-,6-грами	0.35	0.52	0.59	0.91
	LR і словесні 1-грами з синтакс. х-ми	0.36	0.61	0.49	0.92
	LR і буквені 2-,3-,4-,5-,6-грами з синтакс. х-ми	0.39	0.55	0.59	0.89

Слід зауважити, що в таблиці наведені найкращі результати, коли були відібрані найкращі ознаки. Так, для наборів де присутні n-грами були обрано від 1% до 10% з усих ознак для n-грам, при чому чим більше n, тим менший відсоток ознак було взято. Такие розподіл дозволяє позбавити вибірку від неінформативних ознак, та врегулювати відношення типів ознак у випадках їх комбінування.

Як видно з результату, це допомогло отримати більш інформативні дані про роботу кожного з класифікаторів.

3.3. Дослідження моделі на тестових даних

Наступним кроком ми можемо використати справжні тестові дані. В якості вибірки для тестування було взяти тестову вибірку з вище вказаного порталу «Kaggle Data Science» [32]. Дані мають той же формат, що і для навчання. Проте

тепер нам відомі правильні відповіді. Так як набір достатньо великий (більше двох тисяч коментарів), у нас є можливість більш точно оцінити показники класифікації.

Очевидно, що метрики *Accuracy* недостатньо для оцінки якості класифікатору, тому розрахуємо інші показники, визначені в підрозділі 3.4. Також до аналізу та тестування було додано найпростіший варіант для класифікації Baseline – алгоритм, що відносить усі об'єкти до найбільш популярного класу, що присутній в навчальних даних. Ця модель не була описана вище та звісно не претендує до практичного застосування. Даний підхід додано лише для порівняння з іншими.

Так як вибірка достатньо нерівномірна (близько 26% коментарів відносяться до образливих, решта – не образливі), в даному випадку показник *Macro_F1* являється більш об'єктивним, що видно по показникам. В даному випадку судити по показнику *Accuracy* дуже не інформативно, хоча він показує високі результати.

Алгоритм з використанням синтаксичних признаків показав практично той же самий результат, що і *BaseLine* алгоритм. Цей факт свідчить про те, що отримані числові представлення текстів неінформативні і оптимальною поверхнею виявилась та, по одну сторону від якої опинились всі вибірки.

Для другого набору використовувались 2-3-, 4-, 5-, 6-грами, а також їх сукупність. Було проведено дослідження впливу значення *n* на точність класифікації. В усіх випадках значення точності *Accuracy* не дуже високі. Це зумовлено перевагою текстів без образ над текстами з такими. Значення *Macro_F1* із збільшенням *n* зростає та при *n*=6 стає найбільшим. Те саме стосується й інших метрик. Тобто кількість текстів, хибно віднесених до категорії образливих, скорочується як і кількість тих, що були пропущені й залишились не розпізнаними як такі.

Для третього комплекту було взято три окремі моделі з словесними 1-, 2-, та 3-грамами. Кожна з них показує не погані результати, що видно з показника *Macro_F1* та *Accuracy*. Очевидно, що в силу малої кількості слів такі моделі дають гірші показники, ніж у випадку застосування їх для класифікації документів, де обсяг тексту значно перевищує обсяг тексту в навчальній вибірці. Отримані

результати наближені до випадків використання буквених сукупності 2-3-, 4-, 5-, та 6-грам, що підтверджує вищесказане.

Для четвертого комплекту характеристик було взято комбінацію буквених та словесних *n*-грам. В ході тестування та поступового додавання, було виявлено, що використання окремих характеристик в ролі звичайних 1-, 2-, 3-грам та буквених 4-, 5-,6-грам дало незначний приріст показника *Macro_F1*, що в нашому випадку являється більш об'єктивним. Також *Accuracy* показник зазнав приросту.

Додаткове використання синтаксичних характеристик з попереднім підходом покращило метрику *Accuracy* , проте не значно знизило іншу. Це говорить що про погіршення розпізнавальної здатності моделі. Однією з причин цього являється ускладнення моделі кількістю показників, що впливають в кінцевому випадку на результат. Тому вирішено було не враховувати синтаксичні характеристики та залишити третій варіант як базовий.

Для оцінки якості класифікації використовувались метрики *MacroPrecision, MacroRecall, Macro_F1, Accuracy*. Описані вище результати та отримані значення занесені до таблиці 3.3 та представлені відповідно по варіантам модифікації запропонованого методу.

Таблиця 3.3 – Результати показників *Macro_P, Macro_R, Macro_F1* та *Accuracy* для реалізованих моделей

№ гр.	Алгоритм	Метрика			
		Macro_P	*Macro_R*	*Macro_F1*	*Accuracy*
0	BaseLine	0.42	0.50	0.46	0.82
1	LR і синт-ні характеристики	0.45	0.52	0.48	0.84
	LR і буквені 2-,3-грамами	0.81	0.50	0.62	0.53
	LR і буквені 4-грами	0.75	0.57	0.65	0.55
	LR і буквені 5-грами	0.72	0.58	0.64	0.59
	LR і буквені 6-грами	0.66	0.72	0.69	0.61

	LR і буквені 2-,3-,4-,5-,6-грами	0.72	0.55	0.62	0.6
	LR і словесні 1-грами	0.58	0.59	0.58	0.55
	LR і словесні 2-грами	0.69	0.61	0.65	0.64
	LR і словесні 3-грами	0.65	0.59	0.62	0.52
	LR і звичайні 1-,2-,3-грами та буквені 2-,3-,4-,5-,6-грами	0.79	0.72	0.75	0.66
	LR і звичайні 1-,2-,3-грами та буквені 4-,5-,6-грами	0.74	0.81	0.77	0.63
	LR і словесні 1-грами з синтакс. х-ми	0.70	0.62	0.66	0.68
	LR і буквені 2-,3-,4-,5-,6-грами з синтакс. х-ми	0.67	0.64	0.65	0.66

Виходячи з результатів найоптимальніші показники дає 4-й варіант моделі по всім метрикам. Показник точності *Accuracy* не відображає істинної спроможності класифікатора, а інші допомогають краще зрозуміти якість класифікації. Так, варіант «LR і звичайні 1-,2-,3-грами та буквені 4-,5-,6-грами» (далі будемо називати його «Базова модель» або просто БМ для спрощення) у порівнянні з іншим варіантом в тій же категорії, дає незначне погіршення загальної точності *Accuracy,* проте повнота та точність визначення образливих повідомлень – кращі. Тож даний варіант вибрано за базовий для подальшого дослідження.

3.4 Дослідження впливу додаткової обробки тексту на якість класифікації

Так як точність показники роботи модуля на даному етапі не досить високі, як для такого роду систем було вирішено знайти додаткові рішення, що потенційно зможуть покращити результати розпізнавання.

Як вже було сказано на початку даного розділу в підрозділі про результат дослідження предметної області, вхідний текст піддавався лише базовій обробці, яка включала в себе переведення усіх літер до нижнього регістру та видалення знаків пунктуації.

Перше, що повинно гарно вплинути на результат, це використання стемінгу в перед обробці текстів. Стемінг - це процес скорочення слова до основи шляхом відкидання допоміжних частин, таких як закінчення чи суфікс. Результати стемінгу іноді дуже схожі на визначення кореня слова, але його алгоритми базуються на інших принципах. Тому слово після обробки алгоритмом стемінгу (стематизації) може відрізнятися від морфологічного кореня слова. Це допоможе відкинути зайві символи, з яких власне і будуть утворюватися n-грами, а такі символи практично не несуть ніякого змістового навантаження.

Наступним кроком було вирішено видалити зайві службові символи та випадкові символи, так як вхідний набір текстів має певні хеш теги, помарки і тд. Regex (регулярні вирази) підхід було використано для виконання розбору фраз та викорінення спеціальних символів.

Так як кожна мова має певні скорочення слів, англійська мова (мова тестових даних) не є виключенням. Так, наприклад, слово «won't» має таке ж саме значення як і пара «will not». При утворенні n-грам дані відмінності можуть впливати на значення вектору характеристик. Тому доцільно було б виконувати певну нормалізацію вхідного тексту на випадок таких скорочень. Regex (регулярні вирази) підхід було використано для виконання розбору фраз та викорінення спеціальних символів, а також і для нормалізації, що описана вище.

Очевидним наступним потенційним етапом модернізації є використання списку поганих слів (badwords list), що повинно вірогідно принести успіх та покращення показників. Такий список був частково взятий із результаті пошуку на різних ресурсах за допомогою пошукової системи «гугл». Список використовується досить специфічно та являється списком пар образливих слів, що потенційно можуть бути вживані в текстах, на їх оригінальних варіант. Тож всі образливі слова (в межах списку) спочатку приводяться до їх початкового вигляду.

В ряду покращення моделі, яка зараз не являється базовою, та де не використовуються просто уніграми та елементарна обробка тексту, була спроба зробити експерименти з методами класифікації, які навіть були відкинуті на етапі вибору. Так, початково вибрану логістичну регресію було вирішено замінити на метод опорних векторів. Після чого були вибрані також способи класифікації на основі алгоритмів машинного навчання як «*випадковий ліс*» (Random forest) та «градієнтний бустинг» (Gradient boosting).

Аналіз вибірки для навчання та здоровий сенс наштовхують на те, що більшість образ починається з вказівок на персону - іншого користувача. Так, наприклад, фраз з таким шаблоном досить багато: "you are/you're a/an some_abusive_words", "some_abusive_words like you", "you some_abusive_words". Такі аспекти також підверглись опрацюванню в вигляді додаткової характеристики для тексту. Дану додаткову властивість назвемо «you are» підхід, для зручнішого використання в контексті.

Також було розглянуто варіант використання певного діапазону слів, перетворених в біграми, як додаткової характеристики. Такому варіанту модифікації також було дано назву «long range word pairs» підхід.

Після впровадження запропонованих варіантів, що повинні потенційно покращити були зроблені нові заміри згідно обраних показників для оцінки якості моделі.

Найбільше вплинули на результат модернізації зі стемінгом та із списком слів. Останній мав досить не великий обсяг, та певно не покриває всі образливі слова, що наявні в навчальній вибірці (аналіз образливих слів вибірки не проводився). Він був складений із запозиченням слів та пошуку таких на просторах інтернету.

Серед запропонованих варіантів модернізації були такі, що практично не допомогли покращити кінцевий результат. Так, наприклад використання підходу «long range word pairs», при якому фраза з навчальної вибірки *«i really don't understand your point.\xa0 It seems that you are mixing apples and oranges»* приводилась до масиву з слів, якими вона утворена. Після чого було утворено нові

під масиви, де перший включатиме кожне третє слово починаючи з першого, другий масив включатиме кожне третє слово, починаючи з другого, а третій так само, але починаючи з третього. Після чого масиви знову об'єднуються в єдине речення, проте кожний блок буде відділений сепаратором. Така додаткова характеристика у вигляді біграми не вплинула на результат в кращу сторону.

Забігаючи на перед, так як вхідний текст більше не представлений простим способом, а піддається певним маніпуляціям, було вирішено знову протестувати методи класифікації, та глянути яким чином вони зараз впливатимуть на результат. Для цього до моделі класифікації було додано моделі на основі методу «опорних векторів», на основі «дерева рішень» (було використано модель на основі Gradient Tree Boosting, або як його ще називають Gradient Boosted Regression Trees), а також на основі «лісу випадкових вирішаючих дерев» (Forests of randomized trees). Кінцеві показники виявились дещо гіршими, ніж при використанні логістичної регресії.

Говорячи про вдалі модифікації, що при вибраних нами метриках показали вищі результати, варто сказати, що було використано власноруч написані обрізання слів та нормалізація вхідних текстів. Список поганих слів також складений власноруч. Очевидно, що при використанні готових рішень у вигляді певних бібліотек, результат буде дещо кращим. Проте базовий функціонал, описаний в даній магістерській дисертації вже дає досить високі показники.

Як вже було сказано, точність системи (метрика Macro_P) в межах класу – це частка текстів, які дійсно належать даному класу щодо всіх текстів, які система віднесла до цього класу. Повнота системи (метрика Macro_R) – це частка знайдених класифікатором текстів, що належать класу, щодо всіх текстів цього класу в тестовій вибірці. F-міра (метрика Macro_F1) є хорошим кандидатом на формальну метрику оцінки якості класифікатора. Вона зводить до одного числа дві інших основних метрики: точність і повноту. Маючи в своєму розпорядженні подібний механізм оцінки буде набагато простіше прийняти рішення про те чи являються зміни в алгоритмі змінами в кращу сторону, чи ні.

Далі наведено графіки, на яких зображено зміну показників точності, в залежності від доданих функцій до системи.

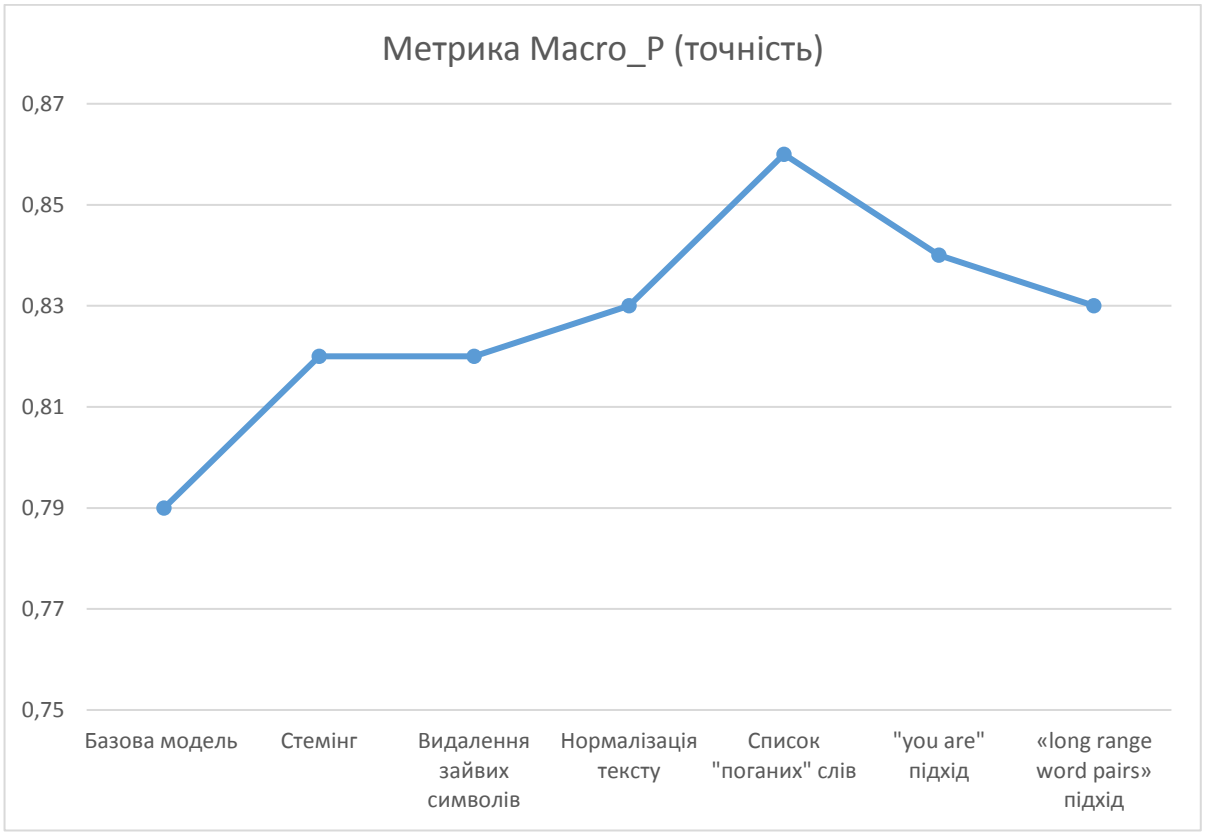

Рисунок 3.2 - Зміна значення метрики Macro_P в залежності від реалізованих додаткових функцій

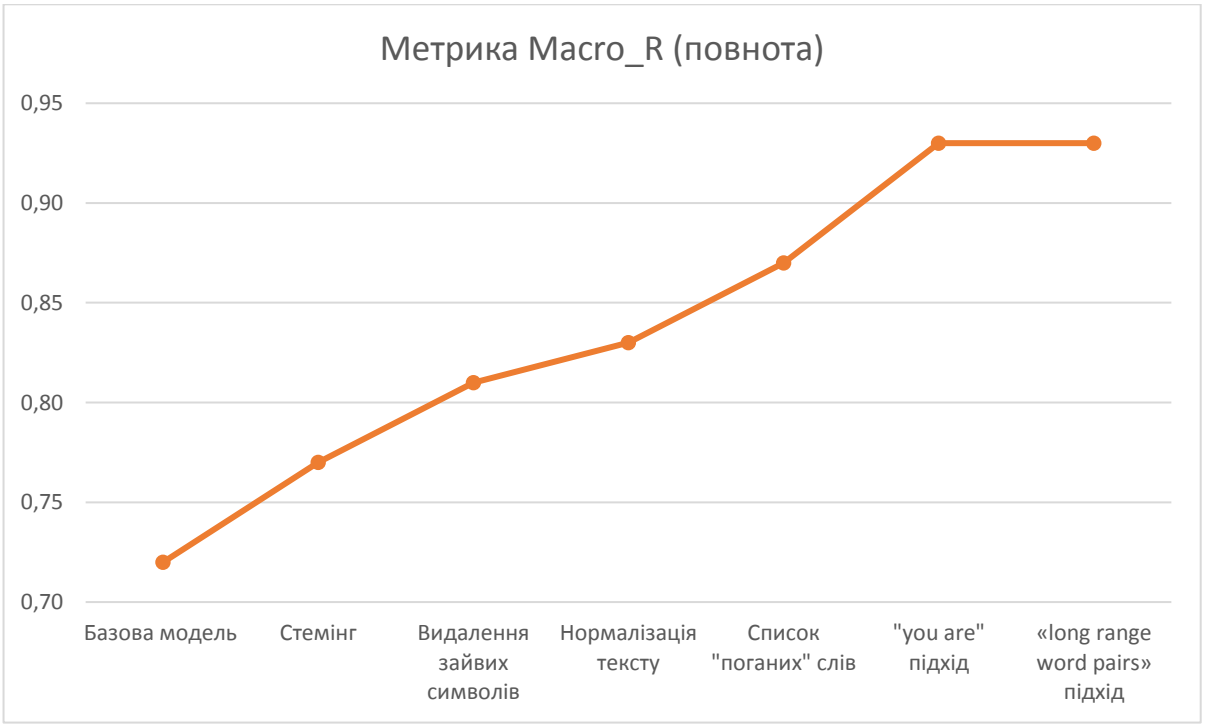

Рисунок 3.3 - Зміна значення метрики Macro_R в залежності від реалізованих додаткових функцій

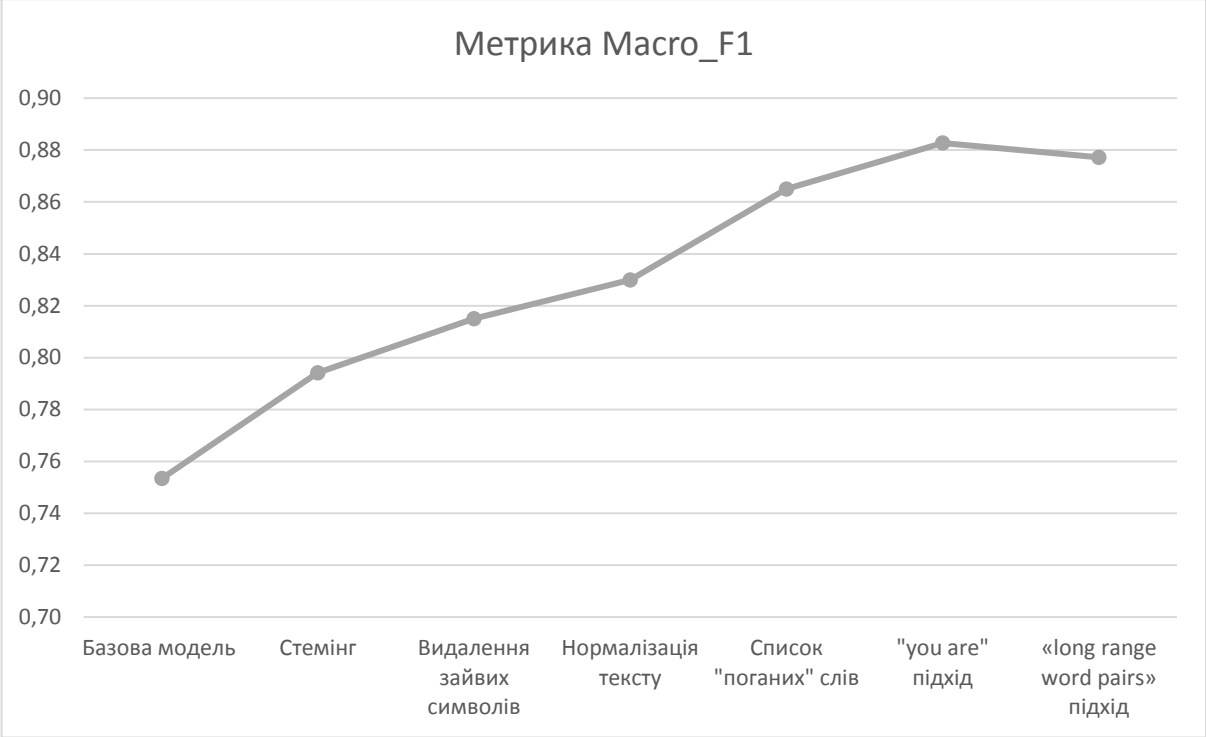

Рисунок 3.4 - Зміна значення метрики Macro_F1 в залежності від реалізованих додаткових функцій

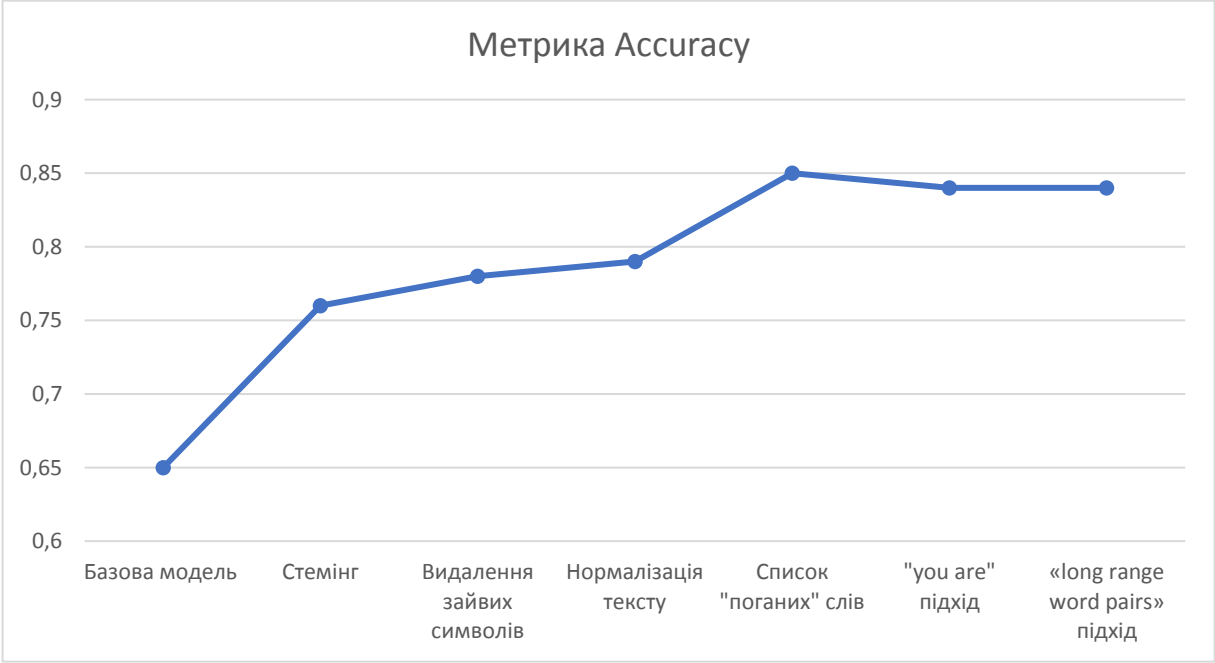

Рисунок 3.5 - Зміна значення метрики Accuracy в залежності від реалізованих додаткових функцій

Тож як бачимо, додані нові функції дозволили покращити показники якості класифікації. Розроблена кінцева модель має можливість для розвитку та покращення, тому показники не являються найвище можливими для даної системи.

3.5 Висновки за розділом

В даному розділі було досліджено вплив параметрів моделі на результати класифікації. Реалізовні моделі з запропонованими наборами характеристик успішно досліджено. За допомогою порівняльного аналізу визначено, що модель «LR і звичайні 1-,2-,3-грами та буквені 4-,5-,6-грами» дала найкращі показники обраних метрик якості та претендує на базову модель для реалізації застосування на її основі.

4 РОЗРОБКА КЛАСИФІКАТОРА НА ОСНОВІ ОБРАНОЇ МОДЕЛІ

Даний розділ присвячений розробці програмного продукту з мінімальним набором функцій, що може бути використаний як окремий модуль для інтеграції в інші системи чи отримати розвиток та стати повноцінним продуктом в подальшому. Далі наведені обгрунтування у виборі засобів розробки для системи класифікації коротких текстів, а також короткий опис роботи класифікатора.

4.1 Аналіз вимог до підсистеми

Розробка будь якої системи та продукту, в тому числі й будь-якого програмного продукту починається зі складання вимог до сутності, що розробляється.

Вимоги до проектованої системи являють собою сукупність тверджень про атрибути, властивості, або якості програмної системи, що підлягають реалізації в майбутньому.

Говорячи конкретно про нашу систему, слід обговорити, що вона повинна вміти для того, щоб досягнути рішення поставлених проблем в даній дипломній роботі.

На вхід системи буде подавитись файл певного формату , або їх набір. На виході система повинна згідно засвоєних навчальних даних робити припущення про приналежність тестового тесту до категорії образливих текстів чи нормальних.

Для кращого формулювання вимог розділимо їх на функціональні та нефункціональні. До перших віднесемо ті, що стосуються конкретно набору функцій, власне алгоритмів роботи та поведінки системи. До останніх віднесемо ті, що стосуються характеру поведінки системи (бізнес-правила, атрибути якості, зовнішній вигляд і тд). Дана система повинна являтись практичним підтвердженням дієздатності висунутої гіпотези, тому набір вимог для неї буде мінімальним.

До категорії функціональних вимог віднесемо:

- Можливість зчитування файлу в форматі CSV з локального простору пристрою;
- Можливість відображати проміжні стани програми та/чи статус виконання;
- Можливість відображати час, затрачений на певний етап роботи;
- Можливість відображати результат класифікації у вигляді заповненого файлу;
- Можливість збереження розпізнаного тексту до текстового файлу одного із форматів;
- Можливість подальшої міграції в інші додатки у вигляді окремого модулю;

Нефункціональні вимоги:

- Можливість використання на персональних комп'ютерах з операційною системою Microsoft Windows 8 та вище;
- Програма повинна мати довільний інтерфейс для виводу статусу виконання.

4.2 Вибір засобів розробки та обґрунтування

Для реалізації додатку, що зможе задовільнити поставлені до нього вимоги, необхідно визначитись з мовою програмування, на якій власне і буде він написаний. Постає питання вибору найзручнішої, що в рамках даного проекту виявилось достатньо складно та принесло деякі труднощі, в причину відсутності досвіду в області реалізації додатків, що пов'язані з машинним навчанням. Очевидно, що простіше всього буде використовувати наявні бібліотеки, які вже мають реалізовані методи та підходи в собі, що повинно суттєво скоротити час, що буде затрачений на написання програми.

В цьому розділі буде описаний короткий аналіз та обґрунтування до вибору мови програмування, існуючих бібліотек з готовими рішеннями, вибір середовища

розробки та інших додаткових засобів. Всі висновки з приводу вибору тих чи інших рішень базуватимуться на основі огляду популярності, відгуків, певних кількісних показниках та перспективах розвитку та подальшого потенційного вдосконалення.

4.2.1 Вибір мови програмування

У новачків в Data Science часто виникає питання про те, яку мову програмування вибрати основною - специфічний, створений спеціально для обробки даних «R» [34], або популярний і в інших сферах та універсальний «Python» [35]. Обидві мови підтримуються open-source ліцензіями (на відміну від комерційних інструментів SAS і SPSS або пропрієтарного MATLAB) та традиційно розглядаються як найбільш затребувані. Стрімкий розвиток Data Science призводить до швидкої зміни позицій фаворитів. Далі буде коротко проаналізовано тенденції в протистоянні Python і R на початок 2018 року. Було прийнято рішення не розглядати інші мови програмування такі як Java, C# або інші «.net framework-s languages» в причину відсутності гідних бібліотек-претендентів з безкоштовною ліцензією для використання в додатках, написаних вказаними мовами (детальніше питання вибору бібліотек та інших засобів розробки описано в наступних розділах).

Аналіз пошукових запитів Google Trends і даних Kaggle показують, що мова програмування загального призначення Python, очевидно, більш затребувана ніж R, і за версією Stack Overflow входить в п'ятірку найбільш популярних мов програмування. Тому порівнювати самі по собі Python і R некоректно - потрібно розглядати їх щодо нашої предметної області. Аналізуючи статистику Google Trends з січня 2012-го року по січень 2018 го по запитам про застосування Python і R в машинному навчанні, видно, що протягом останніх двох років інтерес до Python став переважати і на початок 2018 року число запитів суттєво перевершує аналогічні звернення, пов'язані з «R» (рисунок 4.1).

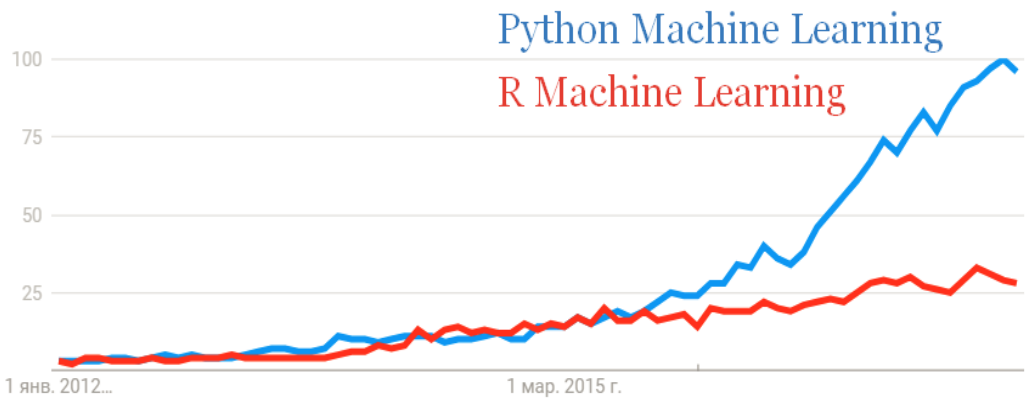

Рисунок 4.1 – Графік попиту на мови програмування Python та R

Виходячи з реальних відгуків спеціалістів, звичайних користувачів та огляду даного сегменту можна виділити переваги та недоліки кожної з мов.

Переваги «R»:

- Мова створювався спеціально для аналізу даних: запис конструкцій мови зрозуміла багатьом фахівцям в області;
- Багато функцій, необхідні для аналізу даних, є вбудованими функціями мови. Перевірка статистичних гіпотез часто займає лише кілька рядків коду;
- Установка IDE (RStudio) і необхідних пакетів обробки даних гранично спрощена;
- Зручний репозиторій пакетів і велика кількість готових тестів практично під всі методи Data Science і машинного навчання;
- Ефективна робота з векторами і матрицями

Недоліки «R»:

- Низька продуктивність;
- Специфічність в порівнянні зі стандартними мовами програмування, так як мова вузькоспеціалізована;
- «R» прекрасний інструмент для статистики і відповідних stand-alone додатків, але просідає в тих областях, де традиційно застосовуються мови загального призначення;

- Є можливість виконати один і той же функціонал різними способами. Синтаксис для розв’язування деяких завдань не зовсім очевидний;
- В силу великої кількості бібліотек, документація більшості, менш популярних з них, не є повною.

Переваги «Python»:

- Практично повністю володіє усіма перевагами мови «R», за деякими винятками;
- Універсальна багатоцільова мова: можна здійснювати не тільки обробку даних, але також їх пошук і використання результату обробки навіть у веб-додатку;
- Є однією з тих мов, які відмінно підходять на роль першої мови програмування. У разі втрати інтересу до машинного навчання чи Data Science набуті навички знадобляться в інших прикладних областях - як при використанні самого Python, так і при освоєнні споріднених мов.

Недоліки «Python»:

- Відсутність загального сховища та брак альтернатив для багатьох бібліотек R. Однак ситуація значно покращилася за останні роки: аналітики, які брали раніше кілька різних мов для різних завдань, відзначають, що набір інструментів змістився за останні два роки в бік бібліотек на Python. Крім того, входження новачків полегшують такі збірки як Anaconda.
- Являється мовою з динамічною типізацією. Це істотно прискорює розробку програм, але заодно ускладнює пошук деяких важко відстежуваних помилок, пов'язаних з неправильним привласненням різних даних одним і тим же змінним.

Отже, з огляду на вище описані аспекти було прийнято рішення використовувати мову Python як більш зрозумілу та перспективну.

4.2.2 Вибір бібліотек для застосування

Для використання у власному додатку якихось готових рішень у вигляді бібліотек, логічним стає пошук таких серед тих, які розповсюджуються за безкоштовною ліцензією. Так як мовою програмування вибрано Python, то остаточний аналіз існуючих бібліотек було проведено після її вибору. Звичайно, початкові обстеження були зроблені під час вирішення питання мови програмування. Далі зроблено огляд на найпопулярніші готові рішення, на які варто звернути увагу.

Для обробки мови та виділення характеристик з текстів було виділено два інструменти.

NLTK (natural language toolkit) - пакет бібліотек і програм для символьного і статистичної обробки природної мови, написаних на мові програмування Python. Містить графічні уявлення і приклади даних. Супроводжується великої документацією, включаючи книгу з поясненням основних концепцій, що стоять за тими завданнями обробки природної мови, які можна виконувати за допомогою даного пакету. NLTK добре підходить для студентів, які вивчають комп'ютерну лінгвістику або близькі предмети, такі як емпірична лінгвістика, когнітивістики, штучний інтелект, інформаційний пошук і машинне навчання. NLTK з успіхом використовується як навчальний посібник, як інструмент індивідуального навчання і в якості платформи для прототипування і створення науково-дослідних систем. NLTK є вільним програмним забезпеченням.

Gensim – популярний інструмент для автоматичної обробки мови, заснований на машинному навчанні і використовуваний як комерційними компаніями, так і академічними дослідниками. У Gensim реалізовані алгоритми дистрибутивної семантики word2vec і doc2vec, він дозволяє вирішувати завдання тематичного моделювання (topic modeling) і виділяти основні теми тексту або документа. Цільова аудиторія є обробка природної мови (НЛП) і IR співтовариство. У Gensim реалізовані популярні алгоритми НЛП, наприклад, word2vec. Більшість реалізацій алгоритмів вміє використовувати кілька ядер.

Для використання готових рішень при роботі в галузі Machine Learning було виділено два інструменти.

Theano - це розширення мови Python, що дозволяє ефективно обчислювати математичні вирази, що містять багатовимірні масиви. Бібліотека надає базовий набір інструментів для конфігурації нейромереж і їх навчання. Найбільше визнання Theano отримала в задачах машинного навчання при вирішенні завдань оптимізації. Бібліотека дозволяє використовувати можливості GPU без зміни коду програми, що робить її незамінною при виконанні ресурсоємних завдань.

TensorFlow - це бібліотека для глибинного навчання, і її зв'язок з Гуглом допомогли проекту TensorFlow залучити багато уваги. Але якщо забути про ажіотаж, деякі його унікальні деталі заслуговують більш глибокого вивчення:

- Основна бібліотека підходить для широкого сімейства технік машинного навчання, а не тільки для глибинного навчання.
- Лінійна алгебра та інші нутрощі добре видно зовні.
- На додаток до основної функціональності машинного навчання, TensorFlow також включає власну систему логування, власний інтерактивний візуалізатор логів і навіть потужну архітектуру з доставки даних.
- Модель виконання TensorFlow відрізняється від scikit-learn мови Python і від більшості інструментів в R.

Scikit-learn - пакет бібліотек для вирішення завдань машинного навчання. Написаний переважно на мові Python, основні алгоритми реалізовано на Cython для досягнення продуктивності. Бібліотека добре документована, містить приклади використання і покрокові інструкції аналізу даних. SciKit - включає засоби для візуалізації даних, що аналізуються і їх оцінки основними популярними методами.

Далі наведено інші загальні бібліотеки, що полегшують процес розробки програмного забезпечення, наприклад, при роботі з масивами, даними і тд.

NumPy - це бібліотека мови Python, що додає підтримку великих багатовимірних масивів і матриць, разом з великою бібліотекою високорівневих математичних функцій для операцій з цими масивами. Математичні алгоритми,

реалізовані на Python, часто працюють набагато повільніше тих же алгоритмів, реалізованих на компільованих мовах (наприклад, Фортран, Сі, Java). Бібліотека NumPy надає реалізації обчислювальних алгоритмів (у вигляді функцій і операторів), оптимізовані для роботи з багатовимірними масивами. В результаті будь-який алгоритм, який може бути виражений у вигляді послідовності операцій над масивами (матрицями) і реалізований з використанням NumPy, працює так само швидко, як еквівалентний код, що виконується в MATLAB.

SciPy - це відкрита бібліотека високоякісних наукових інструментів для мови програмування Python. SciPy містить модулі для оптимізації, інтегрування, спеціальних функцій, обробки сигналів, обробки зображень, генетичних алгоритмів, рішення звичайних диференціальних рівнянь та інших задач, зазвичай вирішуються в науці і при інженерної розробці. Бібліотека розробляється для тієї ж аудиторії, що MATLAB і Scilab. Для візуалізації при використанні SciPy часто застосовують бібліотеку Matplotlib, що є аналогом засобів виведення графіки MATLAB. В даний час SciPy поширюється під ліцензією BSD і його розробники спонсоруються Enthought.

На основі первинного огляду було обрано наступні бібліотеки: SciPy для роботи з матрицями; Scikit-learn – для модуля машинного навчання, а також і для виділення ознак з тексту; Numpy – для додаткових математичних операцій та структур даних.

4.2.3 Вибір середовища розробки

Для Python можна використовувати різні середовища розробки, але однією з найпопулярніших є середовище PyCharm, створена компанією JetBrains. Це середовище динамічно розвивається, постійно оновлюється і доступна для найбільш поширених операційних систем - Windows, MacOS, Linux. На рисунку 4.2 зображений логотип середовища розробки PyCharm.

Рисунок 4.2 - Логотип середовища розробки PyCharm

Правда, вона має одне важливе обмеження. А саме вона доступна в двох основних варіантах: платний випуск Professional і безкоштовний Community. Багато базових можливостей доступні і в безкоштовному випуску Community. У той же час ряд можливостей, наприклад, веб-розробка, доступні тільки в платному Professional.

Іншим середовищем розробки, яке дозволяє працювати з Python, є Visual Studio. Перевагою даної IDE в порівнянні, скажімо, з PyCharm, слід відзначити перш за все те, що в її безкоштовною редакції VS 2017 Community безкоштовно доступні ряд функцій і можливостей, які в тому ж PyCharm доступні тільки в платній версії Professional Edition. У той же час розробка на Python в Visual Studo доступна поки тільки в версії для Windows. На рисунку 4.3 зображений логотип середовища розробки Visual Studio.

Рисунок 4.3 - Логотип середовища розробки Visual Studio

Виходячи з великого досвіду роботи з Visual Studio та більшої міри безкоштовності середовища, було прийнято рішення використати його.

4.2.4 Додаткові засоби розробки

Система контролю версій «Mercurial»

«Mercurial» [36] — вільна розподілена система керування версіями файлів та спільної роботи, розроблена для ефективної роботи з дуже великими репозиторіями сирцевого коду. Mercurial спочатку був написаний для Linux, та пізніше портований під Windows, Mac OS X і більшість Unix-систем. На рисунку 4.4 зображений логотип даної системи.

Рисунок 4.4 – Логотип програми «Mercurial»

Взагалі система є консольною програмою, але існує ще програма під назвою «TortoiseHg» з логотипом у вигляді назви хімічного елемента ртуті (рисунок 4.5), що являється графічною оболонкою для системи керування версіями Mercurial.

Рисунок 4.5 – Логотип програми «TortoiseHg»

«Mercurial» являється програмою, написаною на мові «Python», та на «C». «Mercurial» разом з «TortoiseHg» мають ряд позитивних якостей у використанні:

- Можливість повернення до попередніх версій, як і в інших системах-аналогах;
- Можливість створення необхідної кількості гілок, що дозволяє, наприклад, паралельно розробляти дизайн та логіку;
- Швидкодія;
- Висока продуктивність роботи із сховищем, що не залежить від числа його елементів;

Поряд з традиційними можливостями систем контролю версій, «Mercurial» підтримує повністю децентралізовану роботу (відсутнє поняття основного сховища коду), галуження (можливо вести кілька гілок одного проекту і копіювати зміни між гілками), злиття репозиторіїв (чим і досягається «розподіленість» роботи).

3.8 Короткий опис програми

Для демонстрації функціонування запропонованого підходу та підтвердження гіпотези була написана програма, яка здатна до класифікації коротких текстів по наявності в них образливих виразів та словосполучень. Текст, що подається на вхід додатку після опрацювання буде віднесений до однією з двох категорій: «образливий» та «не образливий». На рисунку 3.8.1 схематично зображена в цілому робота програми, яку можна подати у вигляді послідовності етапів.

Мовою для кодування мною було обрано *Python*, що являється найпопулярнішою для реалізації додатків пов’язаних з нейромережами та машинним навчання. Це дозволило використовувати наявні безкоштовні бібліотеки для підключення, що значно скорочує час на реалізацію алгоритмів.

Основними частинами додатку являються модуль, що визначає та витягує характеристики тексту, що описані раніше, та модуль, що використовує описаний метод для навчання та подальшої класифікації.

Так як програма призвана підтвердити або спростувати дієздатність запропонованого підходу, вона зроблена достатньо просто, та не має дизайнерського інтерфейсу. Далі на рисунку 4.6 зображена схематично програмна реалізація запропонованого метода.

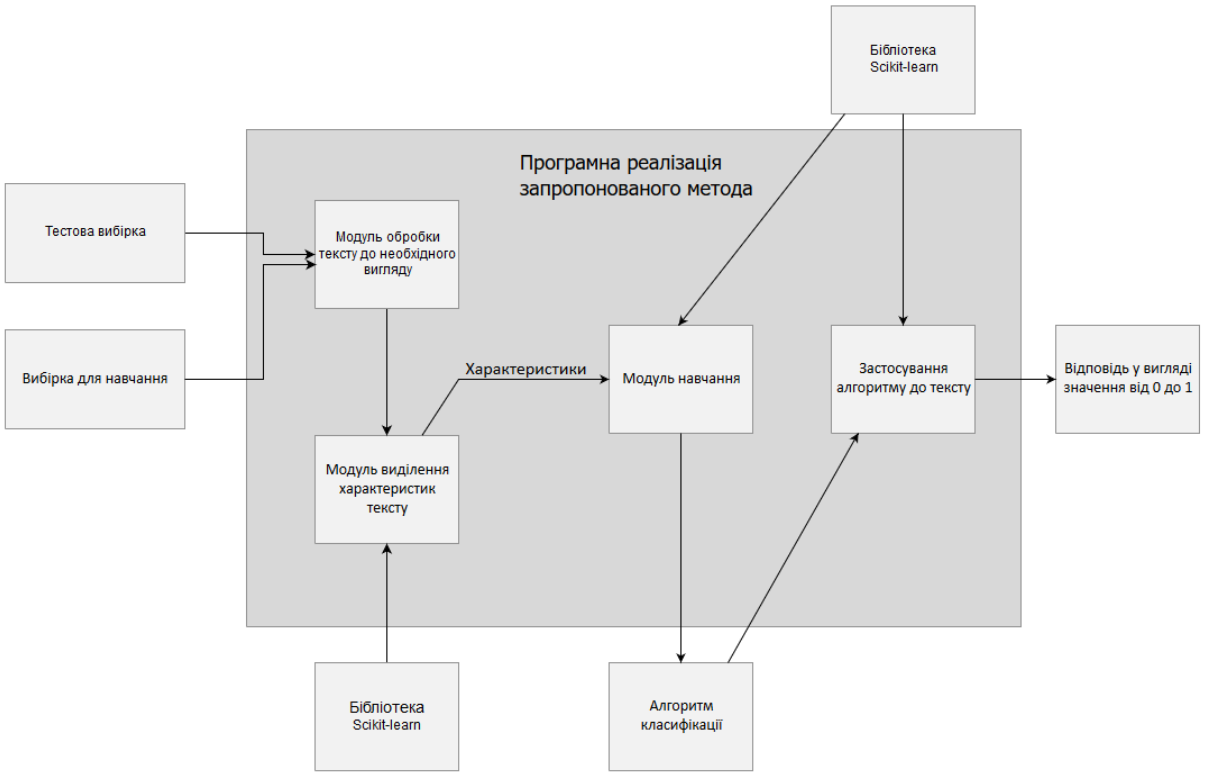

Рисунок 4.6 – Схема роботи програми

Вхідні дані для навчання представлені у вигляді файлу csv (*Comma-Separated Values*) з порталу «Kaggle Data Science» для змагань за напрямом науки про дані. Так як змагання призначені для учасників зі всього світу, всі вирази складені англійською мовою. Кількість записів близько 6500 штук.

Шлях до файлів на даний час прописаний в коді програми. Тому в необхідну папку (зараз це «*C://PersData//Insults//*») необхідно помістити файли з даними для тренування (train.csv, test_with_solutions.csv), файл з «поганими» словами (badwords.txt), файл з тестовими даними (verification.csv) та пустий файл для предикатів (preds.csv).

Файли з даними для навчання повинні бути певного формату: перша колонка з позначками 0\1, що відповідають за приналежність тексту в третій колонці до класу образливих. Дані починаються з другого рядка. Файл для тестування повинен містити тексти в другій колонці. Файл з «поганими» словами повинен містити пари слів в стовпчик, де два слова відокремлюються комою, перше слово являється

потенційно можливим словом в тексті, а друге являється його нормальною формою, оригіналом.

Зчитані дані з файлів далі піддаються обробці. Представлену вибірку необхідно обробити перед проведенням подальшого аналізу. Так наприклад існують випадки, де фраза *“goon it’s about you ”* повинна бути приведена до вигляду *“goon it’s about you”*. Це необхідно для використання більш «чистих» даних для навчання. На рисунку 4.7 зображений фрагмент коду, що видаляє певні зайві знаки з текстів. Було також використано Regex (регулярні вирази) підхід для виконання розбору фраз та викорінення спеціальних символів, а весь текст перетворено до нижнього регістра для спрощення.

```
f = [x.lower() for x in f]
f = [x.replace("\\n"," ") for x in f]
f = [x.replace("\\t"," ") for x in f]
f = [x.replace("\\xa0"," ") for x in f]
f = [x.replace("\\xc2"," ") for x in f]
```

Рисунок 4.7 – Фрагмент коду для очищення тексту від «шумів»

Також текст піддається певній нормалізації. З нього крім «шумів» викоріняються закінчення слів, що не несуть необхідної в даному контексті інформації. Інакше кажучи виконується стемінг (фрагмент коду, що відповідає за це, показано на рисунку 4.8). Для нього не використовувались готові бібліотеки, проте їх використання покриває набагато більше варіантів, що в кінцевому результаті покращить якість системи.

```
f = [re.subn("s( |$)", " ", x)[0].strip() for x in f]
f = [re.subn("ing( |$)", " ", x)[0].strip() for x in f]
f = [x.replace("tard ", " ") for x in f]
```

Рисунок 4.8 – Фрагмент коду для стемінгу для вхідного тексту

Далі формується вибірка для навчання у вигляді набору векторів характеристик для кожного тексту з файлу. Для представлення тексту у векторному

вигляді використано бібліотеку «Scikit-learn», а з її простору *sklearn.feature_extraction.text* було використано об’єкт *TfidfVectorizer,* що використовує статичну міру TF-IDF для оцінки важливості слова в контексті тексту. Об’єкт має метод, що повертає матрицю ознак з векторів ознак тексту. Вказані набори n-грам утворюються на рівні з тестовими даними.

Для машинного навчання, що є наступним етапом, як вже було зазначено, використовується реалізація логістичної регресії. Остання представлена бібліотекою «Scikit-learn», що і була обрана для використання в додатку. Після успішного навчання програма здатна класифікувати тестові тексти за наявністю в них образливих зворотів.

Припущення про приналежність тестових текстів записується до призначеного для цього файлу у вигляді певного числового значення, що лежить в межах від 0 до 1, при чому чим значення ближче до 1, тим більша ймовірність, що відповідний цій мітці текст належить до образливого, і навпаки, якщо значення наближається до 0.

3.9 Висновки за розділом

У даному розділі, що був присвячений розробці програмного продукту з мінімальним набором функцій, проаналізовано існуючі програмні засоби для використання в процесі розробки. Було обгрунтовано їх вибір та описано базову схему роботи застосовання. Мовою реалізації обрано мову Python, а для реалізації необхідних функцій було використано готові рішення у вигляді безкоштовних бібліотек. Отримане застосовання може бути використане як окремий модуль для інтеграції в інші системи з мінімальними змінами. Також його можна вдосконалити до повноцінного комерційного продукту. Наступний розділ присвячений розробці стартап проекту для монетизації та отримання вигоди від розробленого класифікатора.

5 РОЗРОБКА СТАРТАП ПРОЕКТУ

В даному розділі запропонований підхід для монетизації напрацювання, що висвітлене в даній роботі. З огляду на аналіз ринку були сформульовані основні вимоги, описані ідеї, визначені сильні та слабкі сторони потенційного комерційного продукту.

Інформація буде подана у вигляді таблиць.

5.1 Опис ідеї проекту

Головною метою стартап-проекту є розробка універсального модуля, який може бути використаний замовниками за призначенням (про це мова піде далі). Розглянемо зміст ідеї, можливі напрямки застосування, основні переваги, які зможе отримати користувач представлено у таблиці 5.1.

Таблиця 5.1 – Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
	Домашня мережа	Для користувачів-батьків(опікунів) – можливість вберегти дітей від небажаної лексики.
	Інтернет (соціальні мережі)	Для адміністраторів груп в соціальних мережах – автоматичне блокування користувачів в групі, що застосовують нецензурні слова, образи тощо.
	3. Інтернет (різноманітні форуми та портали)	Передбачення розгортання конфліктів на просторах ресурсу мережі між користувачами.

Аналіз ринку показав, що фактично потенційні конкуренти відсутні. Існують вечичезні системи як Proxomitron та HandyCache до складу яких можна або підключити відповідний функціонал, з певними обмеженнями. Перш за все це примітивність, так як підключення відбувається за допомогою скриптів, які писали просунуті користувачі, та котрі можна завантажити з форумів. Висока ступінь примітивізму функціоналу та складність в використанні, обмеженість сфер застосування і тд детальніше описана в таблиці 5.2.

Таблиця 5.2 – Визначення сильних, слабких та нейтральних характеристик ідеї проекту

Техніко-економічні характеристики ідеї	(потенційні) товари/концепції конкурентів			W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
	Мій проект	Proxomitron	Handy Cache			
Мова розробки	C#, Python	-	-			Новітні технології розробки
Тип	Проксі-сервер	Проксі-сервер	Проксі-сервер		Проксі сервер	
Ліцензія	Умовно безкоштовна	Безкоштовна	Умовно безкоштовна		Розширений функціонал або кількість користувачів більше 5 - платна	

Продовження таблиці 5.2 - Визначення сильних, слабких та нейтральних характеристик ідеї проекту

Техніко-економічні характеристики ідеї	Мій проект	Proxomitron	Handy Cache	W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
Засоби для фільтрації	Модель, що побудована на основі засобів штучного інтелекту, в часності машинного навчання.	Скрипти з регулярними виразами, що необхідно завантажувати окремо.	Регулярні вирази, які вставляються користувачем до програми			Застосування засобів штучного інтелекту дозволяє більш коректно розпізнавати не тільки конкретні слова а ідентифікувати завуальовані образи.
Підтримка оновлень	Присутня	Відсутня	Присутня, регулярні оновлення.		Наявність оновлень	
Додатковий функціонал	Відсутній	Присутній	Присутній	Відсутній		

Визначений перелік слабких, сильних та нейтральних характеристик та властивостей ідеї потенційного товару є підґрунтям для формування його конкурентоспроможності.

5.2 Технологічний аудит ідеї проекту

Далі наведений аудит технології, за допомогою якої можна реалізувати ідею проекту.

Визначення технологічної здійсненності ідеї проекту передбачає аналіз таких складових (табл. 5.3):

Таблиця 5.3 – Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1	Графічний інтерфейс	WPF технологія	Наявні готові бібліотеки з відмінними в плані дизайну елементами	Так
2	Модуль прийняття рішення	Модель класифікації на основі методу машинного навчання	Наявні платні та безкоштовні бібліотеки для створення та використання, а також власні доопрацювання	Так
3	Використання готових API сайтів, для адміністрування	Спосіб взаємодії із сторінками сайтів соц. мереж.	Безкоштовний API безпосередньо від ресурсів замовників	Так
Обрана технологія реалізації ідеї проекту: Розробляється програма у вигляді проксі-серверу на мові c# з використанням технології WPF для графічної частини, з модулями на мові «пайтон» для модуля прийняття рішень, що розпізнаватиме в даному контексті небажані вирази(слова). Акцент у виборі засобів для розробки дизайну впав на готові бібліотеки, що для графічного дизайну – безкоштовні. Для побудови модуля прийняття рішень – також безкоштовна, або умовно платна.				

Тож як видно з наведеної вище інформації, технічна можливість реалізації проекту присутня. Складнощів у використанні та отриманні готових бібліотек виникнути не повинно.

5.3 Аналіз ринкових можливостей запуску стартап-проекту

Зробимо аналіз попиту. Далі в таблиці 5.4 наводиться його аналіз, виходячи з наявного ринку.

Таблиця 5.4 – Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	Умовно 2
2	Загальний обсяг продаж, грн/ум.од	-\-
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Відсутні
5	Специфічні вимоги до стандартизації та сертифікації	Практично відсутні
6	Середня норма рентабельності в галузі (або по ринку), %	80%

Ринок достатньо специфічний та вузький, але ніша практично не зайнята. Деякі статистичні дані являються недоступними.

Далі визначені потенційні групи клієнтів, їх характеристики, та сформовано орієнтовний перелік вимог до товару для кожної групи.

В таблиці 5.5 наведено основні характеристики потенційних клієнтів стартап-проекту.

Таблиця 5.5 – Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Вбереження дітей від негативних висловлювань в соц. мережі.	Батьки, навчальні заклади переважно приватного характеру.	Деякі покупці хочуть мати активну статистику використання та моніторингу за активність користувача в соц. мережах.	- до продукції: інтуїтивно зрозумілий інтерфейс налаштування - висока точність ідентифікації нецензурних слів.
2	Потреба блокування користувачів в групах соц. мереж, що активно використовують засоби тролінгу та нецензурні слова.	Адміністратори та власники груп в соціальних мережах.	Деякі покупці захочуть отримати додаткові функції як попередження користувача перший раз, тощо	- до продукції: інтуїтивно зрозумілий інтерфейс налаштування - висока точність ідентифікації нецензурних слів.

Після визначення потенційних груп клієнтів проведено аналіз ринкового середовища: складені таблиці факторів, що сприяють ринковому впровадженню

проекту, та факторів, що йому перешкоджають (таблиці 5.6 та 5.7). Фактори в таблиці подані в порядку зменшення значущості.

Таблиця 5.6 – Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Поява конкурентів	Подальша поява конкурентів після збільшення долі ринку	Розширення функціоналу продукту, та подальший розвиток не тільки як програми «антимат» а і повноцінний фільтр контенту, що надходить до локальної мережі.
2	Ціна товару	Дороговизна розробки провокує високу ціну товару, так як формування ціни ведеться з розрахунком окупності за 3 роки.	Перегляд політки на ціноутворення, можливо зменшення ціни за рахунок залучення більшої кількості користувачі. Продвигання за рахунок реклами.

Таблиця 5.7 – Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Активний піар товару	Продвигання за допомогою якісної реклами.	Користування послугами маркетологів.
2	Максимальна локалізація ПЗ	Розширення ареолу застосування за рахунок інтеграції в широкий спектр,	Розширення списку доступних сайтів для адміністрування та фільтрації висказувань користувачів.

Далі наведено результат проведеного аналіз пропозиції де визначаються загальні риси конкуренції на ринку (таблиця 5.8).

Таблиця 5.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Вказати тип конкуренції - монополія/олігополія/ монополістична/чиста	олігополія	Постійний розвиток та оновлення. Враховування побажань замовників.
2. За рівнем конкурентної боротьби - локальний/національний/…	національний	Максимальна можливість різноманітності локалізацій
3. За галузевою ознакою - міжгалузева/ внутрішньогалузева	міжгалузева	Позитивний вплив
4. Конкуренція за видами товарів: - товарно-родова - товарно-видова - між бажаннями	Товарно-родова. Конкуренція на рівні технології задоволення потреб.	Врахування побажань та їх задоволення приносить позитивний вплив
5. За характером конкурентних переваг - цінова / нецінова	нецінова	Головною конкурентною перевагою є унікальність, адже функціонал достатньо якісний та практичний.

Далі наведено огляд аналізу конкуренції, де проводиться більш детальний аналіз умов конкуренції в галузі (таблиця 5.9).

Таблиця 5.9 – Аналіз конкуренції в галузі за М. Портером

	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
Складові аналізу	PROXOMITRON та HandyCache	Детальний аналіз ринку показує, що потенційні бар'єри для виходу на ринок відсутні	Відсутні, адже постачальники бібліотек та готових рішень не створюють перешкод	Споживачі можуть бути не задоволені роботою деяких функцій та хотіти додати щось нове	Практично відсутні
Висновки:	Практично відсутня конкуренція зі сторін: PROXOMITRON та HandyCache	- Є можливість виходу на ринок; - Потенційні конкуренти є, проте вони не задовольняють потреби ринку; - Строк виходу на ринок близько 15 місяців	Постачальники не диктують умови роботи на ринку	Клієнти вносять побажання, що доцільно задовольняти в певній мірі.	Практично відсутні

Так як конкуренція досить слабка – ринок стає привабливим для освоєння. Проблеми із знаходження частково готових рішень також немає: наявні бібліотеки

претенденти для використання під час розробки додатку (модуля) постачаються або за безкоштовною, або умовно безкоштовною ліцензією. Головними сильними сторонами продукту буде функціональність, якість роботи та зручність використання, що не можна сказати про конкурентів.

На основі аналізу конкуренції, проведеного в таблиця 5.9, а також із урахуванням характеристик ідеї проекту, вимог споживачів до товару та факторів маркетингового середовища визначається та обґрунтовується перелік факторів конкурентоспроможності. Аналіз оформлюється за табл. 5.10.

Таблиця 5.10 – Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
1	Задоволення конкретних потреб	На основі аналізу форумів програм аналогів можна дійти висновку, що потреби користувачів в той чи іншій функції не задоволені, або задоволені лише частково. Тому даний проект призваний задовольнити потреби.
2	Максимальна інтуїтивність взаємодії	
3	Естетичний вигляд	
4	Точність виявлення нецензурних слів	

За визначеними факторами конкурентоспроможності (табл. 5.10) проводиться аналіз сильних та слабких сторін стартап-проекту (табл. 5.11). (С.П. – стартап проект, К.1 – конкурент 1, К.2 – конкурент 2).

Таблиця 5.11 – Порівняльний аналіз сильних та слабких сторін мого проекту

№ п/п	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні						
			-3	-2	-1	0	1	2	3
1	Інтуїтивність інтерфейсу	18				К.1-2		С.П.	
2	Достовірність розпізнавання	20	К.1-2						С.П.
3	Якісна характеристика розпізнавання	20	К.1-2						С.П.
4	Детальність налаштування	18	К.1-2				С.П.		
5	Рівень звітності	20		К.1-2					С.П.
6	Вартість	20		С.П.	К.1	К.2			
	Оптимізованість	20	К1-2						С.П.

Фінальним етапом ринкового аналізу можливостей впровадження проекту є складання SWOT-аналізу (матриці аналізу сильних (Strength) та слабких (Weak) сторін, загроз (Troubles) та можливостей (Opportunities) (табл. 5.12) на основі виділених ринкових загроз та можливостей, та сильних і слабких сторін (табл. 5.11).

Таблиця 5.12 – SWOT- аналіз стартап-проекту

Сильні сторони: Задоволення потреб ринку: • висока якість розпізнавання • високий рівень кастомізації • зручність використання	Слабкі сторони: • ціна товару; • відносна складність розробки; • необхідність кастомізації під кожного замовника;
Можливості: • розширення сфер застосування.	Загрози: • малий обсяг продаж; • поява конкурентів.

На основі SWOT-аналізу розробляються альтернативи ринкової поведінки (перелік заходів) для виведення стартап-проекту на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок (див. табл. 5.9, аналіз потенційних конкурентів).

Визначені альтернативи аналізуються з точки зору строків та ймовірності отримання ресурсів (табл. 5.13).

Таблиця 5.13 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Поліпшення темпів розповсюдження завдяки роботі маркетологів	присутня	стислі
2	Зменшення витрат на розробку завдяки використанню безкоштовних рішень	проста	обширні
3	Короткочасне зменшення ціни на продукт для заманювання клієнтів. Розділення функцій програми на пакети для вибору	проста	стислі

Для початку доцільно обрати альтернативу номер 3.

5.4 Розроблення ринкової стратегії проекту

Розроблення ринкової стратегії першим кроком передбачає визначення стратегії охоплення ринку.

Доцільним буде описати кожен профіль цільової групи потенційних клієнтів. При цьому важливим пунктом для аналізу являється готовність споживачів сприйняти продукт.

Також аналізуємо орієнтовний попит в межах цільової групи, інтенсивність конкуренції в сегменті, а також оцінено простоту входу до сегменту кожної цільової групи потенційних клієнтів.

Опис цільових груп потенційних споживачів (табл. 5.14).

Таблиця 5.14 – Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Батьки	Висока зацікавленість	Середній попит	Практично відсутня на початку. Подальше збільшення	Середня складність
2	Адміністратори\власники груп\пабліків в соц. мережах, що націлені на наукові та офіційні теми.	Ймовірна зацікавленість у груп, що націлені на наукові та офіційні теми.	Середній-вище середнього	Практично відсутня на початку. Подальше збільшення	Складність нижче середнього
3	Офіційні сайти, портали новин…	Зацікавленість вище середньої	Середній попит	Практично відсутня на початку. Подальше збільшення	Висока складність
Краще всього підійдуть групи 1 та 2 так як простота входу та рівень зацікавленості прийнятні.					

За результатами аналізу потенційних груп споживачів (сегментів) автори ідеї обирають цільові групи, для яких вони пропонуватимуть свій товар, та визначають стратегію охоплення ринку:

- якщо компанія зосереджується на одному сегменті – вона обирає стратегію концентрованого маркетингу;
- якщо працює із кількома сегментами, розробляючи для них окремо програми ринкового впливу – вона використовує стратегію диференційованого маркетингу;
- якщо компанія працює із всім ринком, пропонуючи стандартизовану програму (включно із характеристиками товару/послуги) – вона використовує масовий маркетинг.

Тож для роботи в обраних сегментах ринку необхідно сформувати базову стратегію розвитку (табл. 5.15).

Таблиця 5.15 – Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкуренті позиції відповідно до обраної альтернативи	Базова стратегія розвитку
1	Короткочасне зменшення ціни на продукт для заманювання клієнтів. Розділення функцій програми на пакети для вибору (різні в залежності від замовників)	За рахунок розділення функцій на пакети можливе охоплення більшої частини ринку, адже замовник купує тільки те, що йому потрібно.	Так як конкуренти в тому вигляді, що є, не створюють значної конкуренції даний пункт можна залишити пустим	Стратегія диференціації

Наступним кроком є вибір стратегії конкурентної поведінки (табл. 5.16).

Таблиця 5.16 – Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
1	Ні	Так, буде	Копіювання буде лише основних функцій, але із значним вдосконаленням	Стратегія лідера

На основі вимог споживачів з обраних сегментів до постачальника (стартап-компанії) та до продукту (див. табл. 5.5), а також в залежності від обраної базової стратегії розвитку (табл. 5.15) та стратегії конкурентної поведінки (табл. 5.16) розробляється стратегія позиціонування, що полягає у формуванні ринкової позиції (комплексу асоціацій), за яким споживачі мають ідентифікувати торгівельну марку/проект.

В таблиці 5.17 наведена розроблена стратегія позиціонування продукту.

Таблиця 5.17 – Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
1	Базові навички володіння ПК.	Стратегія диферен-ціації	Якісні та кількісні показники розпізнавання Естетичний вигляд та високий рівень кастомізації	Вбереження дітей від бранних слів Чистота коментарів та публікацій на публічних сторінках. Отримання статистики переглядів контенту в мережі іншими особами

Підсумок: Тож компанія розробник вибирає розвиток в напрямку ключових функцій: 1) Убереження дітей замовника від бранних слів в соц мережах та інтернеті. 2) Чистота коментарів та публікацій на публічних сторінках для замовників у вигляді адмінів останніх. 3) Отримання статистики переглядів контенту в мережі іншими особами.

5.5 Розроблення маркетингової програми стартап-проекту

Маркетингова програма стартап-проекту розроблена на основі потреб виходячи з концепції потенційного товару. Сформульовано вигоди та переваги, що пропонує товар, особливо в порівнянні з конкурентами.

Першим кроком є формування маркетингової концепції товару, який отримає споживач. Для цього у таблиці 5.18 потрібно підсумувати результати попереднього аналізу конкурентоспроможності товару.

Таблиця 5.18 – Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити
1	Рівень та достовірність розпізнавання	Найвищий рівень, постійне оновлення бази знань	Найвищій показник.
2	Зрозумілість та зручність	Естетично приємний дизайн, інтуїтивний інтерфейс	Завдяки використанню нових технологій – значна перевага перед конкурентами.

Надалі розробляється трирівнева маркетингова модель товару: уточнюється ідея продукту та/або послуги, його фізичні складові, особливості процесу його надання (табл. 5.19).

Таблиця 5.19 – Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові
І. Товар за задумом	Програма типу проксі-сервер для комерційного чи приватного користування націлена на очищення тексту сторінок від нецезурних слів. Можливе використання на операційній системі Віндовс 7 та вище.

Продовження таблиці 5.19 – Опис трьох рівнів моделі товару

ІІ. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх /Тл/Е/Ор
	1. Період обслуговування за ліцензією	Років	2
	2. Зручність використання	-	Висока
	3. Наявність динамічних підказок	-	Так
	4. К-сть параметрів налаштування	Шт.	30+
	5. Кількість доступних сайтів та соц. мереж для адміністрування	Шт.	1000+
	6. Кількість мов підтримки	Шт.	5
	7. Формування звітів.	-	Так
	Програма пройшла усі можливі тестування та повністю відповідає визначеним головним вимогам ринку.		
	Постачається в електронному вигляді на сайті виробника.		
	Марка: Попко-лимитед. Назва: Проект-альфа		
	Програмне забезпечення		
	Програмне забезпечення		
Проект буде захищено від копіювання регістрацією назви програми, створення заявки на отримання патенту на винахід, щоб уберегти алгоритм роботи від копіювання.			

Після формування маркетингової моделі товару слід особливо відмітити – чим саме проект буде захищено від копіювання. Захист може бути організовано за рахунок захисту ідеї товару (захист інтелектуальної власності), або ноу-хау, чи комплексне поєднання властивостей і характеристик, закладене на другому та третьому рівнях товару.

Наступним кроком є визначення цінових меж, якими необхідно керуватись при встановленні ціни на потенційний товар (остаточне визначення ціни

відбувається під час фінансово-економічного аналізу проекту), яке передбачає аналіз ціни на товари-аналоги або товари субститути, а також аналіз рівня доходів цільової групи споживачів (табл. 5.20). Аналіз проводиться експертним методом.

Таблиця 5.20 – Визначення меж встановлення ціни

№ п/п	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
	Замінники відсутні	Аналог у вигляді програми HandyCache пропонує функцію фільтрації за окремі гроші в складі безкоштовної програми за у вигляді розширення за 1199грн. Та ніякого натяку на функція адміністрування.	Нижче середнього - Середні та вище середнього	Так як програма стає практично монополістом, то нижній рівень складатиме 1499,89грн. Верхня межа-8999,89грн. Для індивідуальних замовників – ціна договірна.

Наступним кроком є визначення оптимальної системи збуту, в межах якого приймається рішення:

- проводити збут власними силами або залучати сторонніх посередників (власна або залучена система збуту);
- вибір та обґрунтування оптимальної глибини каналу збуту;
- вибір та обґрунтування виду посередників;
- вибір оптимальної с-ми збуту

Далі наведено аналіз та вибір оптимальної системи збуту (табл. 5.21).

Таблиця 5.21 – Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	Придбання ключів-активаторів для програми	Електронний вигляд. Доступ для купівлі через сайт	Виробник-споживач	Офіційний сайт виробники з конфігуратором та сторінкою придбання

Розроблення концепції маркетингових комунікацій (табл. 5.22).

Таблиця 5.22 – Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1	Занижена ціна на початку компанії з продажу. Пропозиція скористатись всіма перевагами на протязі тріал-періоду в 15 днів	Лінія зворотного зв’язку на сайті. Онлайн-чат.	Система відгуків,	Переконати в необхідності придбання такого ПЗ .	Інтернет реклама від гугл, що буде відображатись тільки в містах, де шукають таку інформацію.

Тепер результатом стає програма, що включає в себе концепції товару, збуту, просування та попередній аналіз можливостей ціноутворення, спирається на

цінності та потреби потенційних клієнтів, конкурентні переваги ідеї, стан та динаміку ринкового середовища, в межах якого буде впроваджено проект, та відповідну обрану альтернативу ринкової поведінки.

5.5 Висновки за розділом

Вказана програма типу проксі-сервер для комерційного чи приватного користування націлена на очищення тексту сторінок від нецезурних слів. Вона включає в себе версії для комерційного та приватного користування. Розширений функціонал дозволяє проводити функції адміністрування, пов’язані з блокуванням, попередженням користувачів-порушників та\або редагуванням негативних коментарів. Основна задача для батьків вберегти дітей від негативу. Основа ціль для адмінів – автоматично фільтрувати розміщення постів та коментарів на сайті для підтримання чистоти висловлювання та збереження статусу.

Можливе використання даного ПЗ на операційній системі Віндовс 7 та вище.

Так як конкуренція досить слабка – ринок стає привабливим для освоєння. Проблем із знаходження частково готових рішень для розробки - немає. Технології, що були обрані – знаходяться у вільному доступі, або умовно платні, за помірну ціну. Головними сильними сторонами буде функціональність, якість роботи та зручність використання, що не можна сказати про конкурентів.

Очікуваними бар’єрами для входження буде висока ціна. Тож для швидшого входження на ринок вибрано підходи з використанням рекламних заходів, а також використання зменшення ціни на короткий час. Потенціальні користувачі отримають можливість відчути усі переваги ПЗ за допомогою тріал-версії на 15 днів.

Потенційні або прямі конкуренти можуть з’явитись одразу після очікуваного успіху входження на ринок. В такому випадку доцільно буде продовжити розвиток, та задовольнити побажання та відгуки замовників, щоб збільшити кількість придбання ПЗ користувачами.

Проект буде захищено від копіювання регістрацією назви програми, створення заявки на отримання патенту на винахід, щоб уберегти алгоритм роботи від копіювання.

ВИСНОВКИ ТА РЕКОМЕНДАЦІЇ

В даній магістерській дисертації були представлені результати дослідження методів і засобів для розв’язання задачі на визначення змісту короткого тексту (повідомлення, коментарю) за наявністю в ньому образ. В рамках даних досліджень були розглянуті деякі існуючі способи класифікації математичними методами. Запропоновано модернізовані моделі для досягнення позитивного результату, а також визначено деякі можливі їх модифікації для подальшого визначення кращого в рамках поставленої мети. Досліджено характеристики текстів, а саме буквені n-грами, словесні n-грами, ряд синтаксичних признаків, а також їх поєднання.

На основі тестування було обрано модель на основі логістичної регресії (Logistic Regression), так як на базових характеристиках текстів вона показала найкращий результат. Найгірші результати дає модель на основі виключно ряду синтаксичних характеристик.

Результати дослідження показали, що дійсно, найпопулярніші моделі на основі словесних n-грам також мають право на прикладне застосування в задачі бінарної класифікаціїї на коротких текстах. Проте в причину малої кількості ознак потребують перегляду чи певних модернізацій.

Було визначено п’ять наборів категорій ознак для текстів. На основі порівняння метрик класифікатора було обрано найоптимальніший. Ним виявився набір їз звичайними словесними 1-,2-,3-грамами та буквеними 4-,5-,6-грамами у вигляді окремих n-грам. Використання синтаксичних характеристик, як додаткових ознак не дало значного покращення, а в деяких випадках знижувало точність класифікації.

Основною складністю, з якою довелось зіткнутись виявилось подолання явища перенавчання класифікатора на кожній моделі, так як гарантовано найкращого рішення не існує, а пошук такого потребує аналізу поведінки системи в цілому.

На основі обраної оптимальної моделі реалізовано програмне забезпечення, що здатне виділити характеристики тексту, що описують його емоційну складову

та образливі звороти. Попередньо текст піддається обробці для видалення певного «шуму», що не несе ніякої інформації, чи являється службовим текстом і тд. Запропоновані методи і програмні засоби пройшли тестові випробування, в тому числі й на реальних по об’єму та змісту коментарях.

В якості майбутніх напрямків досліджень та вдосконалення моделі вибрано наступні:

- Розгляд та дослідження методів автоматичного складання словників для слів, що явно буду вказувати на образу. Знайдене слово або комбінація таких дозволить практично з найвищою ступеню ймовірності віднести даний коментар (текст) до категорій образливих. З урахуванням специфіки коментарів в соціальних мережах є сенс враховувати друкарські помилки при пошуку таких. Як варіант, їх необхідно буде порівнювати та визначати ступінь схожості з вже існуючими. Дана система отримала лише базову версію такого вдосконалення.
- Можливість ідентифікувати сарказм. Прикладом цього може бути коментар типу: «Продовжуй! Продовжуй! Коли ти говориш, я відчуваю себе таким розумним!». В даному виразі не має прямих образ, проте прихований зміст ображає того, кому він адресований. На даний момент реалізація моделі, що описана в цій роботі не здатна навчитись ідентифікувати ці аспекти. Проте за допомогою NLP проектів, таких як «Stanford CoreNLP» можна зрозуміти структуру висловлювання та в певній мірі його зміст.
- Розширення кількості характеристик тексту та присвоєння їм необхідного ступеню значущості, що в певній мірі повинно покращити якість класифікації текстів.

СПИСОК ЛІТЕРАТУРИ

1. The advanced theory of language as choise and chance / Herdan Gustav // Kommunikation und Kybernetik in Einzeldarstellungen. – 1966. – С. 14–437.
2. The Statistical Study of Literary Vocabulary / Udny Y. G. // Shoe String Press Inc. U.S. – 1968.
3. Shakespeare, fletcher, and the two noble kinsmen / Ledger G., Merriam T. // Literary and Linguistic Computing. – 1994. – С.235–248.
4. Общий взгляд на машинное обучение: классификация текста с помощью нейронных сетей и TensorFlow [Електронний ресурс] – Режим доступу: https://tproger.ru/translations/text-classification-tensorflow-neural-networks. – (дата звернення 03.03.2017). – Назва з екрана.
5. A mechanical solution of a literary problem. / The Popular Science Monthly, c.97-105. [Електронний ресурс] / Mendenhall T. C. // Popular Science Monthly. – 1901. – №. 60. – Режим доступу: https://en.wikisource.org/wiki/Popular_Science_Monthly/Volume_60/December_1901/A_Mechanical_Solution_of_a_Literary_Problem. – (дата звернення 10.04.2017). – Назва з екрана.
6. Isaiah and the computer: A preliminary report [Електронний ресурс] / Radday Y. // Computers and the Humanities. – 1970. – №. 5. – С. 65–73. – Режим доступу: https://link.springer.com/article/10.1007/BF02402282. – (дата звернення 16.04.2018). – Назва з екрана.
7. Authorship attribution [Електронний ресурс] / Juola P. // Foundations and Trends in Information Retrieval. – 2006. – № 1. – С. 233—334. – Режим доступу: https://books.google.com.ua/books/about/Authorship_Attribution.html?id=_B2zDLdqe60C&redir_esc=y. – (дата звернення 11.04.2017). – Назва з екрана.
8. N-gram-based author profiles for authorship attribution [Електронний ресурс] / Keselj V., Peng F., Cercone N., Thomas C. – Режим доступу:

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.9.7388&rep=rep1&type=pdf. – (дата звернення 11.04.2017). – Назва з екрана.

9. Scientific and Engineering Problem-Solving with the Computer / Bennett W. R. // Prentice Hall Series in Automatic Computation. – 1976. – №1.
10. Author identification: Using text sampling to handle the class imbalance problem [Електронний ресурс] / Stamatatos E. // Information Processing and Management: an International Journal. – 2008. – № 44. – С.790–799. – Режим доступу: https://dl.acm.org/citation.cfm?id=1347518. – (дата звернення 18.03.2018). – Назва з екрана.
11. The battle of The Quiet Don: Another pilot study [Електронний ресурс] / Kjetsaa G. // Computers and the Humanities. – 1977. – №11. – С.341-346. – Режим доступу: https://www.jstor.org/stable/30205004?seq=1#page_scan_tab_contents. – (дата звернення 16.04.2018). – Назва з екрана.
12. Quantitative authorship attribution: An evaluation of techniques [Електронний ресурс] / Grieve J. // Literary and Linguistic Computing. – 2005. – №22. – С.251-270. – Режим доступу: https://academic.oup.com/dsh/article-abstract/22/3/251/951481?redirectedFrom=fulltext. – (дата звернення 01.04.2018). – Назва з екрана.
13. Experiments with mood classification in blog posts [Електронний ресурс] / Mishne G. // 1st workshop on stylistic analysis of text for information access. – 2005. – Режим доступу: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.111.2693&rep=rep1&type=pdf. – (дата звернення 10.04.2018). – Назва з екрана.
14. Mining newsgroups using networks arising from social behavior [Електронний ресурс] / Agrawal R., Rajagopalan S., Srikant R., Xu Y. // Proceedings of the 12th international conference on World Wide Web. – 2003. – С. 529–535. – Режим доступу:

https://dl.acm.org/citation.cfm?id=775227. – (дата звернення 03.05.2017). – Назва з екрана.

15. Twitter sentiment analysis: The good the bad and the omg! [Електронний ресурс] / Kouloumpis E., Wilson T., Moore J. // Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media. – 2011. – С. 538–541. – Режим доступу: http://www.aaai.org/ocs/index.php/ICWSM/ICWSM11/paper/download/2857/3251. – (дата звернення 10.09.2017). – Назва з екрана.
16. Rule-based approach to sentiment analysis at ROMIP 2011 [Електронний ресурс] / Kan D. // AlphaSense Inc.– 2011. – Режим доступу: http://www.dialog-21.ru/media/1393/138.pdf. – (дата звернення 16.11.2017). – Назва з екрана.
17. Определение авторства тексту с использованием буквенной и грамматическй информации [Електронний ресурс] / Кукушкина О.В., Поликарпов А. А., Хмелев Д. В. // Проблемы передачи информации. – 2001. – №2. – С. 96–109. – Режим доступу: http://www.philol.msu.ru/~lex/articles/grco_r.htm. – (дата звернення 01.04.2017). – Назва з екрана.
18. TF-IDF [Електронний ресурс] – 2018. – Режим доступу: https://uk.wikipedia.org/wiki/TF-IDF. – (дата звернення 02.03.2018). – Назва з екрана.
19. Naive Bayes classifier [Електронний ресурс] – 2018. – Режим доступу: https://en.wikipedia.org/wiki/Naive_Bayes_classifier. – (дата звернення 02.03.2018). – Назва з екрана.
20. Support vector machine [Електронний ресурс] – 2018. – Режим доступу: https://en.wikipedia.org/wiki/Support_vector_machine. – (дата звернення 02.03.2018). – Назва з екрана.
21. Logistic regression [Електронний ресурс] – 2018. – Режим доступу: https://en.wikipedia.org/wiki/Logistic_regression. – (дата звернення 02.03.2018). – Назва з екрана.

22. Рекомендательные системы: теорема Байеса и наивный байесовский классификатор [Електронний ресурс] / Николенко С. // Блог компании Surfingbird. – 2012. – Режим доступу: https://habr.com/company/surfingbird/blog/150207/ – (дата звернення 18.08.2017). – Назва з екрана.
23. Рекомендательные системы: SVD, часть I [Електронний ресурс] / Николенко С. // Блог компании Surfingbird. – 2012. – Режим доступу: https://habr.com/company/surfingbird/blog/139863/ – (дата звернення 17.08.2017). – Назва з екрана.
24. Bag-of-words model [Електронний ресурс] – 2018. – Режим доступу: https://en.wikipedia.org/wiki/Bag-of-words_model. – (дата звернення 16.09.2017). – Назва з екрана.
25. Leave-one-out cross-validation [Електронний ресурс] – 2018. – Режим доступу: https://en.wikipedia.org/wiki/Cross-validation_(statistics)#Leave-one-out_cross-validation. – (дата звернення 02.03.2018). – Назва з екрана.
26. Support function [Електронний ресурс] – 2018. – Режим доступу: https://en.wikipedia.org/wiki/Support_function. – (дата звернення 16.09.2017). – Назва з екрана.
27. Лекции по логическим алгоритмам классификации [Електронний ресурс] / Воронцов К. В. – 2010. – С. 17-21. – Режим доступу: http://www.machinelearning.ru/wiki/images/3/3e/Voron-ML-Logic.pdf – (дата звернення 17.09.2017). – Назва з екрана.
28. Персептроны [Електронний ресурс] / Минский М., Пейперт С. – 1971. – Режим доступу: https://sheba.spb.ru/delo/perseptrony-1969.htm – (дата звернення 17.09.2017). – Назва з екрана.
29. Laplacian smoothing [Електронний ресурс] – 2018. – Режим доступу: https://en.wikipedia.org/wiki/Laplacian_smoothing. – (дата звернення 02.03.2018). – Назва з екрана.

30. Proxomitron [Електронний ресурс] – 2018. – Режим доступу: https://ru.wikipedia.org/wiki/Proxomitron. – (дата звернення 02.03.2018). – Назва з екрана.
31. HandyCache [Електронний ресурс] – 2018. – Режим доступу: https://ru.wikipedia.org/wiki/HandyCache. – (дата звернення 02.03.2018). – Назва з екрана.
32. Kaggle. Your Home for Data Science [Електронний ресурс] – 2018. – Режим доступу: https://www.kaggle.com/. – (дата звернення 02.03.2018). – Назва з екрана.
33. Using kullback-leibler distance for text categorization [Електронний ресурс] / Bigi B. // Proceedings of the 25th European conference on IR research. – 2003 – С. 305–319. – Режим доступу: https://goo.gl/igNQ5P – (дата звернення 11.01.2018). – Назва з екрана.
34. R (programming language) [Електронний ресурс] – 2018. – Режим доступу: https://en.wikipedia.org/wiki/R_(programming_language). – (дата звернення 02.03.2018). – Назва з екрана.
35. Python (programming language) [Електронний ресурс] – 2018. – Режим доступу: https://ru.wikipedia.org/wiki/Python. – (дата звернення 02.03.2018). – Назва з екрана.
36. Mercurial SCM [Електронний ресурс] – 2018. – Режим доступу: https://www.mercurial-scm.org/. – (дата звернення 10.03.2018). – Назва з екрана.
37. Математические методы обучения по прецедентам (теория обучения машин) [Електронний ресурс] / Воронцов К. В. // Курс лекций «Машинное обучение» – Режим доступу: http://www.machinelearning.ru/wiki/images/6/6d/Voron-ML-1.pdf. – (дата звернення 11.09.2017). – Назва з екрана.

Додаток А

Наукова стаття

УДК 004.855.5

ДОРОГИЙ Я.Ю.,
ДЕМЧИНСЬКИЙ В.В.,
ПОПКО А. В.

ДОСЛІДЖЕННЯ МЕТОДІВ ВИЗНАЧЕННЯ ЗАБАРВЛЕННОСТІ ТЕКСТОВИХ ПОВІДОМЛЕНЬ

В даній статті розглянуто питання вибору математичних засобів для рішення задачі ідентифікації коментарів з соціальних мережах як образливих та реалізації програмного модулю по обраній моделі.

This article discusses the selection of mathematical tools for solving the problem of identifying comments from social networks as offensive and implementing a software module for the selected model.

Ключові слова: класифікація текстів, машинне навчання, метод опорних векторів.

1. Вступ

Обсяг тексту коментарів до публікацій в соціальних мережах практично завжди перевищує безпосередньо сам запис. Часто серед висловлювань користувачів можна зустріти досить негативні, що часто супроводжуються нецензурною лексикою та образливими виразами, які в подальшому можуть призводити до розростання конфлікту на місці публікації. Модерація коментарів вручну чи їх відключення частково вирішує проблему зі сторони публікації. Якщо ж стоїть питання фільтрації та відключення зі сторони користувача, наприклад батьківський контроль, то вирішення являється дещо складнішим. Наявність програмного модуля, що зміг би сортувати вхідний текст відносно невеликого обсягу принаймні на дві категорії за наявністю образ чи їх відсутності, дало б змогу автоматизувати процес ідентифікації та в подальшому створити різноманітні обгортки для використання в різних умовах.

2. Постановка мети роботи

Метою розробки та аналізу, які представлені в даній статті являється програмний комплекс, на вхід якого можна подати текст, що очевидно являється коментарем. На виході модуль повинен повернути клас до якого належить дане висловлювання: «образливе» чи «не образливе». Це стосується будь-якої мови, тому доцільно розробити продукт, що здатний до налаштування під різні мови.

3. Огляд та аналіз існуючих рішень

В більшості випадків кожен з алгоритмів для вирішення даної задачі класифікувати двома складовими: за вибором характеристик тексту, що корелюють з його забарвленням та по методу їх обробки. Далі наводяться деякі характеристики та методи.

Характеристики тексту

Характеристика тексту, як правило, представляє собою деяку величину, яку є можливість обрахувати для даного тексту. Як приклад можна розглядати розподілення довжин слів у виразі. Існує велика кількість задач аналізу тексту, що потребують попереднього визначення його характеристик. До їх числа відноситься, наприклад, задача визначення авторства тексту. Ефективність використання даної характеристики оцінювалась експериментально.

В якості характеристик тексту для визначення його авторства в різні часи пропонувалось використовувати наступні [1–3]:

- Розподіл довжин слів, середня довжина слова [5-7];
- Розподіл сполучень символів деякої довжини (n – буквені n-грами) [8, 9]
- Розподіл частин тексту та їх позиції в реченні. [1-3]. Розглядався розподіл частин на декількох перших позиціях та в кінці речення;
- Розподіл часто вживаних в мові слів. Існують частотні словники, де вказується кількість зустрічань певного слова на мільйон інших в колекції текстів; [8, 9]
- Словарний запас [10, 11]. В роботі [12] також запропоновані показники для тексту як $V_1 = \frac{V}{L}$; $V_2 = \frac{V}{\sqrt{L}}$; $V_3 = \frac{\log V}{\log L}$; $V_4 = \frac{\log V}{\log\log L}$, де V – число унікальних слів в тексті, L – їх загальна кількість в тексті.

Пізніше виявилось, що більшість з них підходять і для рішення задачі на визначення емоційного забарвлення та наявність образливих зворотів в тексті.

Існують також дещо інші характеристики, що можна використовувати, якщо мова йде коментарі з соціальних мереж:

- Наявність слів, написаних повністю заголовними літерами[13];
- Наявність слів, що оточені різними астерисками (знаками) [13];
- Наявність смайлів, посилань, хеш тегів тощо [13–15].

Методи обробки характеристик тексту

Одними із перших методів для визначення емоційного забарвлення тексту, були методи, що засновані на певних правилах (Rule-based methods). Їх робота полягає в пошуку в тексті характерних синтаксичних сполучень, конструкцій, що можуть з високою ймовірність указати на емоційне забарвлення чи наявність образи. Наприклад, вживання частки «ні» перед позитивно забарвленими словами, що характеризують особистість тощо, змінює значення на негативне. Більш складні правила можуть враховувати структуру речення, наприклад, наявність заперечних часток чи сполучників в складносурядних чи складнопідрядних реченнях: «…ти молодець, але (проте)...». Якість таких методів напряму залежить від якості та кількості запропонованих правил та потребує роботи лінгвіста. Автоматичний синтаксичний розклад речення являється досить складною задачею. Приклад даного підходу розглянуто в роботі [16]. Прості елементи даного підходу використовують разом в сполученні з другими методами, наприклад, машинного навчання.

В деяких випадках в задачах класифікації текстів використовують різноманітні статистичні методи і критерії. В роботі [17] текст розглядають як Марковський ланцюг символів. В роботі [12] запропоновано Хі-квадрат – статистику як міру визначення схожості текстів. Теоретично, такі методи можна використовувати і для рішення задачі визначення образ в коротких текстах (коментарях), проте надійних експериментальних результатів в даному напрямку поки що немає.

Найчастіше на практиці використовують методи машинного навчання для досягнення результату поставленої задачі. На відміну від методів, заснованих на правилах, вони дозволяють враховувати більшу кількість характеристик. Даний підхід дозволяє абстрагуватись від прив'язаності до конкретної мови та являється автоматизованим.

Метод опорних векторів

Нехай $Y = \{1, -1\}$. Метод опорних векторів будує класифікатор наступного вигляду:

$$a(x, w, w_0) = \text{sign}\left(\sum_{i=1}^{n} w_i \xi_i - w_0\right).$$

де $w = (w_1, \ldots, w_n) \in \Re^n$, $w_0 \in \Re$ – параметри алгоритму. Рівняння $\langle w, x\rangle = w_0$ описує гіперповерхню в просторі $\Re^n$. Ідея методу заключається в побудові оптимальною в деякому розумінні гіперповерхні, що буде розділяти класи $y =$

1 та $y = -1$. Вибірка лінійно подільна, якшо існує така гіперповерхня, що об'єкти різних класів находяться по різні сторони даної поверхні. Оптимальною називається така розділяюча гіперповерхня, що відстань від неї до найближчого об'єкта максимальна у порівнянні с іншими розділяючими поверхнями. Ці вимоги записуються системою як:

$$\begin{cases} \langle w, w \rangle \to \min; \\ y_i(\langle w, x_i \rangle - w_0) \geqslant 1, \ \ i = 1 \dots l. \end{cases}$$

Вибірки на практиці зазвичай являються лінійно нероздільні. Метод узагальнюється на цей випадок шляхом дозволу похибок на об'єктах. В такому випадку вимоги будуть записуватись системою як:

$$\begin{cases} 0.5\langle w, w \rangle + C \sum_{i=1}^{k} e_i \to \min; \\ y_i(\langle w, x_i \rangle - w_0) \geqslant 1 - e_i, \ \ i = 1 \dots l; \\ e_i \geqslant 0, \ \ i = 1 \dots l. \end{cases}$$

Тут e_i – розмір похибки на об'єкті x_i, C – параметр алгоритму, оптимальне значення якого знаходиться експеременталько і регулює співвідношення між шириною смуги і сумарною похибкою на об'єктах.

У випадку коли вибірка лінійно неподільна зазвичай використовується «ядровий трюк» (*kernel trick*). Його суть наступна. Нехай дано відображення $\varphi: \Re^n \to H$, де H – простір із скалярним добутком $\langle a_1, a_2 \rangle_H$. Якщо цей простір має більш високу розмірність, то вельми ймовірно, що вибірка в ньому буде роздільна. Рівняння розділяючої гіперповерхні в просторі H буде мати вигляд $\langle \varphi(X), w \rangle H = w_0$.

Відповідь на питання «Якою буде поверхня та як буде виглядіти в просторі» залежить від H. На практиці беспесередньо відображення нас не цікавить, достатньо знати ядро – функцію $K(x_1, x_2) = \langle \varphi(x_1), \varphi(x_2) \rangle_H$.

Приклади ядер:

- $K(x_1, x_2) = \langle (x_1), (x_2) \rangle_{\Re^n}$ – лінійне ядро, розділяюча поверхня матиме вид гіперповерхні;
- $K(x_1, x_2) = (\langle (x_1), (x_2) \rangle_{\Re^n})^d$ – поліноміальне ядро, розділяюча поверхня буде мати вид поверхності порядку d;
- $K(x_1, x_2) = e^{\frac{|x_1 - x_2|^2}{2\sigma^2}}$ – гауссове ядро с параметром σ;
- $K(x_1, x_2) = th(k_0 + k_1 \langle (x_1), (x_2) \rangle_{\Re^n})$ – сигмоїдне ядро с параметрами k_0, k_1.

Існують правила, за якими можемо побудувати свої ядра, проте в більшості випадків на практиці використовують одні з вказаних вище ядер. Для модифікації алгоритму с урахуванням ядер достатньо всюди в програмі зробити заміну звичайного скалярного добутку на функцію ядра.

Даний метод опорних векторів широко використовують в різноманітних задачах машинного навчання, в яких він показує гарні результати. Можливість варіювати ядро K и параметр C забеспечує методи необхідну гнучкість. В загальному випадку тимчасова складність алгоритма достатньо висока, а саме поліноміальна від вибірки. Для лінійного ядра існують методи, що дозволяють найти близьку до оптимальної гіперповерхню за лінійний проміжок часу.

4. Запропонований метод рішення

В даній роботі, результати якої будуть наведені далі, була висунута гіпотеза, що використання буквених *n*-грам як характеристик тексту, в купі з методами машинного навчання може дати гарний кінцевий результат при класифікації тексту за наявність образливих оборотів, так як це може бути в коментарях до записів з соціальних мереж. До такого висновку наштовхує те, що одним із найчастіше вживаних методів для класифікації текстів являється використання звичайних *n*-грам в поєднанні з будь-яким методом машинного навчання. Зазвичай в ролі останнього використовують метод опорних векторів SVM (*support vector machine*). Проте, в силу властивостей та специфіки соціальних мереж, вхідний текст має переважно невелику кількість слів. Це негативно впливає на якість розпізнавання та віднесення до певного класу при використанні *n*-грам. Тексту ж буде відповідати велика кількість буквенних *n*-грам. Останні успішно використовуються

для класифікації текстів по їх авторству [12], коли їх поєднують з використанням статичних критеріїв. Далі було розглянуті наступні варіанти:

- Кожному тексту ставиться у відповідність чисельний вектор характеристик, що буде складатись з частоти зустрічання в ньому буквених *n*-грам при $n \leqslant 3$. Або ж індикаторів наявності буквених *n*-грам при $n > 3$.
- Для уникнення великої кількості характеристик є варіант використання відстані між розподіленнями. Для цього необхідно для кожного заданого n розрахувати розподілення буквених n-грам на даному тексті. Після цього розраховується відстань від даного розподілення до розподілення буквених n-грам на множині текстів з позитивним забарвленням і відсутністю в ньому образ та з фразами, що їх мають. В якості відстані були взяті наступні функції:

$$L_1(\mathrm{p},\mathrm{q}) = \sum_i |p_i - q_i|;$$
$$L_2(\mathrm{p},\mathrm{q}) = \sum_i (p_i - q_i)^2;$$
$$D_{KL}(\mathrm{p},\mathrm{q}) = \sum_i \ln\left(\frac{p_i}{q_i}\right) p_i;$$

Тут p, q – дискретний розподіл з ймовірністю i-го значення p_i і q_i відповідно, D_{KL} – відстань Кульбака-Лейблера між двома розподілами, яке використовувавсь в тому числі при класифікації текстів [18]. Таким чином, якщо використовувати m відстаней, то даному тексту при даному n у нас буде можливість поставити у відповідність $2m$ відстаней: m відстаней до розподілу n-грам на множині текстів, що не мають образ та m відстаней до множини текстів з їх присутністю.

Для отримання кращого результату при обчисленні характеристик використовувались тільки *n*-грам, які зустрічаються в наборі коментарів більше заданого числа разів. Даний підхід дозволяє враховувати наявність друкарських помилок в тексті.

Також, в даній задачі було запропоновано використовувати ряд синтаксичних характеристик тексту, які успішно використовуються в рішеннях задач по визначенню авторства тексту [10]. Виходячи з чого було складено наступні комплекти характеристик:

1. Синтаксичні:
   - Довжина речення в словах;
   - Середня довжина слова в символах;
   - Середня довжина виразів (речень) в словах;
   - Середня довжина виразів (речень) в символах;
   - Розподіл кількості слів заданої довжини (від 1 до 20);
   - Розподілення частин мови;
   - Розподілення частин мови, що стоять на першій та другій позиціях в реченні.
2. Буквені *n*-грами. Їх можна розглядати як набір характеристик при одному *n*, так і союз (поєднання) декількох наборів характеристик при різних *n*. Так, наприклад, доцільним буде розглядати 2-грами та 3-грами разом в силу їх відносно невеликої кількості. В такому випадку їм буде відповідати набір характеристик, що скрадатиметься із частоти зустрічання буквених 2-, 3-грам.
3. Відстань між розподіленнями буквених *n*-грам в даному тексті і розподіленні на множині всіх текстів одного класу.

5. Оцінка якості

Для визначення ефективності роботи алгоритму необхідно мати декілька метрик, що відображатимуть якість віднесення висловлювань до однієї з двох категорій на тестовому наборі фраз:

tp_x – кількість об'єктів класу *x*, що віднесені алгоритмом до класу *x*

fp_x – кількість об'єктів не класу *x*, що віднесені алгоритмом до класу *x*;

fn_x – кількість об'єктів класу *x*, що віднесені алгоритмом не до класу *x*;

tn_x – кількість об'єктів не класу *x*, що віднесені алгоритмом не до класу *x*;

S – множина усіх класів, до котрих відносяться об'єкти вибірки.

Для основних метрик якості було запропоновано наступні:

- Precision – частка об'єктів, що класифіковані алгоритмом до

конкретного класу, та які дійсно відносяться до нього:

$$P = \frac{tp_x}{tp_x + fp_x}$$

- Macro Precision – середнє значення для точності по усім класам:

$$\text{Macro\_P} = \frac{1}{|S|} \sum_{x \in S} \frac{tp_x}{tp_x + fp_x}$$

- Recall – частка об'єктів конкретного класу, що вірно класифікована алгоритмом і віднесена до нього:

$$R = \frac{tp_x}{tp_x + fn_x}$$

- Macro Recall – середнє значення по повноті серед усіх класів:

$$\text{Macro\_R} = \frac{1}{|S|} \sum_{x \in S} \frac{tp_x}{tp_x + fn_x}$$

- F1-measure – середнє гармонічне для метрик Precision та Recall:

$$F1 = \frac{2PR}{P + R}$$

- Macro F1-measure – середнє по F1-measure серед усіх класів:

$$\text{Macro\_F1} = \frac{1}{|S|} \sum_{x \in S} F1_x$$

- Accuracy – частка вірно класифікованих об'єктів серед усіх об'єктів по усім класам:

$$\text{Accuracy} == \frac{1}{|S|} \sum_{x \in S} \frac{tp_x + tn_x}{tp_x + fn_x + tn_x + fp_x}$$

6. Короткий опис роботи програми

Для демонстрації функціонування запропонованого підходу та підтвердження гіпотези була написана програма, яка здатна до класифікації коротких текстів по наявності в них образливих виразів та словосполучень. Текст, що подається на вхід додатку після опрацювання буде віднесений до однією з двох категорій: «образливий» та «не образливий». На мал.1 схематично зображена в цілому робота програми, яку можна подати у вигляді послідовності етапів. Мовою для кодування мною було обрано *Python*, що являється найпопулярнішою для реалізації додатків пов'язаних з нейромережами та машинним навчання. Це дозволило використовувати наявні безкоштовні бібліотеки для підключення, що значно скорочує час на реалізацію алгоритмів.

Основними частинами додатку являються модуль, що визначає характеристики тексту, що описані раніше, та модуль, що використовує метод опорних векторів для навчання та подальшої класифікації.

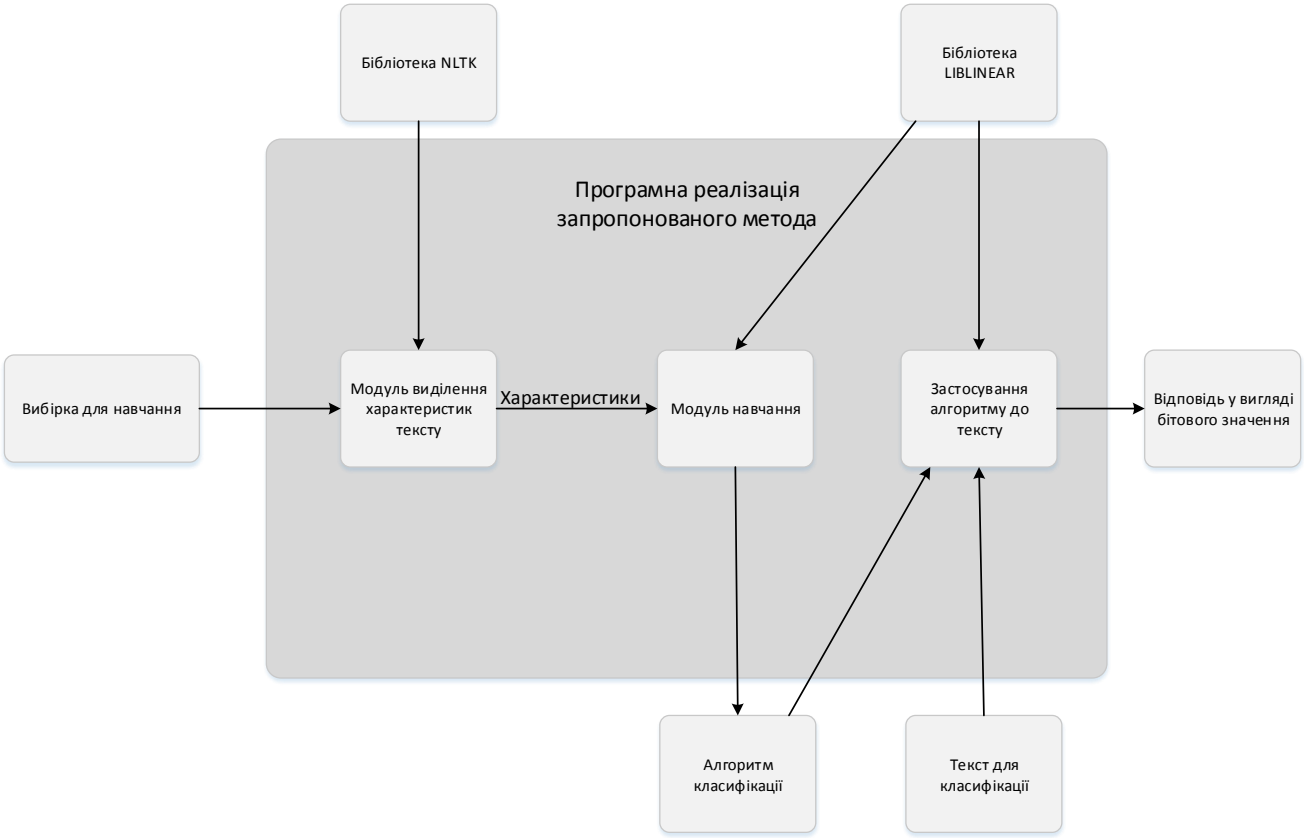

Рис. 1. Схема роботи програми

Вхідні дані для навчання представлені у вигляді файлу csv (*Comma-Separated Values*) з порталу «Kaggle Data Science» для змагань за напрямом науки про дані. Так як змагання призначені для учасників зі всього світу, всі вирази складені англійською мовою. Кількість записів близько 4000 штук. Представлену вибірку необхідно обробити перед проведенням подальшого аналізу. Так наприклад існують випадки, де фраза *"goon it's about you "* повинна бути приведена до вигляду *"goon it's about you"*. Це необхідно для використання більш «чистих» даних для навчання. Regex (регулярні вирази) підхід було використано для виконання розбору фраз та викорінення спеціальних символів, весь текст перетворено до нижнього регістра для спрощення.

Наступним етапом роботи програми являються аналіз текстів. Тут обраховуються характеристики кожного запису з вибірки та середні характеристики для класу. Для виділення характеристик тексту було використано бібліотеку під назвою NLTK (*Natural Language Toolkit*), що написана на мові Python. Вона підходить для морфологічного, семантичного аналізу, стемінгу (скорочення слова до основи), лематизації (покращений варіант стемінгу) і тд на різних мовах. Розповсюджується за ліцензією Apache.

Далі формується вибірка для навчання у вигляді набору векторів характеристик для кожного тексту з файлу.

Для машинного навчання, що є наступним етапом, як вже було зазначено, використовується реалізація методу опорних векторів. Остання представлена бібліотекою «Liblinear», що і була обрана для використання в додатку. Після успішного навчання програма здатна класифікувати інший текст за наявністю в ньому образливих зворотів.

7. Результати роботи та тестування

Для даної задачі визначення чи являються тексти, що подаються на вхід образливими чи ні, були протестовані набори характеристик, що були описані в розділі «Запропонований метод». Для другого

набору використовувались 2-,3-, 4-, 5-, 6-грами. Для третього були протестовані 2- і 3-грами.

В якості вибірки для тестування було взяти тестову вибірку з вище вказаного порталу «Kaggle Data Science». Дані мають той же формат, що і для навчання. Проте тепер нам відомі правильні відповіді. Так як набір достатньо великий (близько двох тисяч коментарів), у нас є можливість більш точно оцінити показники класифікації.

Для оцінки якості класифікації використовувались метрики *MacroPrecision, MacroRecall, Macro_F1, Accuracy*. В табл. 1 представлені дані відповідно по варіантам модифікації запропонованого методу.

Найпростішим варіантом для класифікації було взято Baseline – алгоритм, що відносить усі об'єкти до найбільш популярного класу. Так як вибірка достатньо нерівномірна (близько 26% коментарів відносяться до образливих, решта – не образливі), в даному випадку показник *Macro_F1* являється більш об'єктивним, що видно по показникам.

Алгоритм з використанням синтаксичних признаків показав той самий результат, що і *BaseLine* алгоритм. Цей факт свідчить про те, що отримані числові представлення текстів неінформативні і оптимальною поверхнею виявилась та, по одну сторону від якої опинились всі вибірки.

Було проведено дослідження впливу значення *n* на точність класифікації. В усіх випадках значення точності *Accuracy* не дуже високі. Це зумовлено перевагою текстів без образ над текстами з такими. Значення *Macro_F1* із збільшенням *n* зростає та при *n*=6 стає найбільшим. Те саме стосується й інших метрик. Тобто кількість текстів, хибно віднесених до категорії образливих, скорочується як і кількість тих, що були пропущені й залишились не розпізнаними як такі.

Табл. 1. Результати тестування

Алгоритм	Метрика			
	Macro_P	Macro_R	Macro_F1	Accuracy
BaseLine	0,422	0,50	0,47	0,84
SVM і синтаксичні характеристики	0,422	0,50	0,47	0,84
SVM і відстань між розподілами буквенних 2-,3-грам	0,830	0,55	0,58	0,53
SVM і буквенні 2-,3-грамами	0,81	0,53	0,61	0,53
SVM і буквенні 4-грами	0,80	0,56	0,58	0,55
SVM і буквенні 5-грами	0,62	0,69	0,61	0,69
SVM і буквенні 6-грами	0,66	0,72	0,67	0,79

8. Висновок

В даній роботі представлені результати дослідження методів і засобів для вирішення задачі на визначення змісту короткого тексту (повідомлення, коментарю) за наявністю в ньому образ. В рамках даних досліджень були розглянуті деякі існуючі способи класифікації математичними методами. Запропоновано власний варіант для досягнення позитивного результату, а також визначено деякі можливі його модифікації для подальшого визначення кращого в рамках поставленої мети. Запропоновано і досліджено характеристики текстів, а саме буквені *n*-грами та ряд синтаксичних признаків. Реалізовано програмне забезпечення, що здатне визначати характеристики тексту, що описують емоційну складову та образливі звороти. Попередньо текст піддається обробці для видалення певного «шуму», що не несе ніякої інформації, чи являється службовим текстом і тд. Зроблено опис методу машинного навчання «Метод опорних векторів» та показаний спосіб його застосування для аналізу отриманих характеристик. Запропоновані методи і програмні засоби пройшли тестові випробування, в тому числі й на реальних по об'єму та змісту коментарях.

В якості майбутніх напрямків досліджень та вдосконалення моделі вибрано наступні:

- Розгляд та дослідження методів автоматичного складання словників

для слів, що явно буду вказувати на образу. Знайдене, так зване «стоп-слово» або комбінація дозволить практично з найвищою ступеню ймовірності віднести даний коментар (текст) до категорій образливих. З урахуванням специфіки коментарів в соціальних мережах є сенс враховувати друкарські помилки при пошуку таких. Як варіант, їх необхідно буде порівнювати та визначати ступінь схожості з вже існуючими. Існує вже досить багато словників, складені за подібними тематиками. Один з найбільших представляється компанією «Google» та доступний для вільного користування.

- Можливість ідентифікувати сарказм. Прикладом цього може бути коментар типу: «Продовжуй! Продовжуй! Коли ти говориш, я відчуваю себе таким розумним!». В даному виразі не має прямих образ, проте прихований зміст ображає того, кому він адресований. На даний момент реалізація моделі, що описана в цій роботі не здатна навчитись ідентифікувати ці аспекти. Проте за допомогою NLP проектів, таких як «Stanford CoreNLP» можна зрозуміти структуру висловлювання та в певній мірі його зміст.
- Розширення кількості характеристик тексту, що в певній мірі повинно покращити якість класифікації текстів.

Список літератури

1. The advanced theory of language as choise and chance / Herdan Gustav // Kommunikation und Kybernetik in Einzeldarstellungen. – 1966. – С. 14–437.
2. The Statistical Study of Literary Vocabulary / Udny Y. G. // Shoe String Press Inc. U.S. – 1968.
3. Shakespeare, fletcher, and the two noble kinsmen / Ledger G., Merriam T. // Literary and Linguistic Computing. – 1994. – С.235–248.
4. Общий взгляд на машинное обучение: классификация текста с помощью нейронных сетей и TensorFlow [Електронний ресурс] – Режим доступу: https://tproger.ru/translations/text-classification-tensorflow-neural-networks.
5. A mechanical solution of a literary problem. / The Popular Science Monthly, c.97-105. [Електронний ресурс] / Mendenhall T. C. // Popular Science Monthly. – 1901. – №. 60. – Режим доступу: https://en.wikisource.org/wiki/Popular_Science_Monthly/Volume_60/December_1901/A_Mechanical_Solution_of_a_Literary_Problem.
6. Isaiah and the computer: A preliminary report [Електронний ресурс] / Radday Y. // Computers and the Humanities. – 1970. – №. 5. – С. 65–73. – Режим доступу: https://link.springer.com/article/10.1007/BF02402282
7. Authorship attribution [Електронний ресурс] / Juola P. // Foundations and Trends in Information Retrieval. – 2006. – № 1. – С. 233—334. – Режим доступу: https://books.google.com.ua/books/about/Authorship_Attribution.html?id=_B2zDLdqe60C&redir_esc=y
8. N-gram-based author profiles for authorship attribution [Електронний ресурс] / Keselj V., Peng F., Cercone N., Thomas C. – Режим доступу: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.9.7388&rep=rep1&type=pdf.
9. Scientific and Engineering Problem-Solving with the Computer / Bennett W. R. // Prentice Hall Series in Automatic Computation. – 1976. – №1.
10. Author identification: Using text sampling to handle the class imbalance problem [Електронний ресурс] / Stamatatos E. // Information Processing and Management: an International Journal. – 2008. – № 44. – С.790–799. – Режим доступу: https://dl.acm.org/citation.cfm?id=1347518.
11. The battle of The Quiet Don: Another pilot study [Електронний ресурс] / Kjetsaa G. // Computers and the Humanities. – 1977. – №11. – С.341-346. – Режим доступу: https://www.jstor.org/stable/30205004?seq=1#page_scan_tab_contents.
12. Quantitative authorship attribution: An evaluation of techniques [Електронний ресурс] / Grieve J. // Literary and Linguistic Computing. – 2005. – №22. – С.251-270. – Режим доступу: https://academic.oup.com/dsh/article-abstract/22/3/251/951481?redirectedFrom=fulltext.
13. Experiments with mood classification in blog posts [Електронний ресурс] / Mishne G. // 1st workshop on stylistic analysis of text for information access. – 2005. – Режим доступу: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.111.2693&rep=rep1&type=pdf
14. Mining newsgroups using networks arising from social behavior [Електронний ресурс] / Agrawal R., Rajagopalan S., Srikant R., Xu Y. // Proceedings of the 12th international conference on World Wide Web. – 2003. – С. 529–535. – Режим доступу: https://dl.acm.org/citation.cfm?id=775227.
15. Twitter sentiment analysis: The good the bad and the omg! [Електронний ресурс] / Kouloumpis E., Wilson T., Moore J. // Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media. – 2011. – С. 538–541. – Режим доступу: http://www.aaai.org/ocs/index.php/ICWSM/ICWSM11/paper/download/2857/3251.
16. Rule-based approach to sentiment analysis at ROMIP 2011 [Електронний ресурс] / Kan D. // AlphaSense Inc.– 2011. – Режим доступу: http://www.dialog-21.ru/media/1393/138.pdf.

17. Определение авторства тексту с использованием буквенной и грамматическй информации [Електронний ресурс] / Кукушкина О.В., Поликарпов А. А., Хмелев Д. В. // Проблемы передачи информации. – 2001. – №2. – С. 96–109. – Режим доступу: http://www.philol.msu.ru/~lex/articles/grco_r.htm.
18. Using kullback-leibler distance for text categorization [Електронний ресурс] / Bigi B. // Proceedings of the 25th European conference on IR research. – 2003 – С. 305—319. – Режим доступу: https://ai2-s2-pdfs.s3.amazonaws.com/f9c7/4b45203266abf92f2f40e4b268aaf3274d38.pdf.
19. Математические методы обучения по прецедентам (теория обучения машин) [Електронний ресурс] / Воронцов К. В. // Курс лекций «Машинное обучение» – Режим доступу: http://www.machinelearning.ru/wiki/images/6/6d/Voron-ML-1.pdf.

Додаток Б
Теза

Insults detection on short texts

Andrii Popko
Department of ACTS, FICT
NTUU "Igor Sikorsky Kyiv Polytechnic Institute"
Kyiv, Ukraine
andrew.popko.j@gmail.com

Yaroslav Dorohyi
Department of ACTS, FICT
NTUU "Igor Sikorsky Kyiv Polytechnic Institute"
Kyiv, Ukraine
email@gmail.com

Abstract. **Theses of the report contain a description of the choice of a mathematical model for solving the problem of classification of short texts for the presence of insults in them. The paper briefly describes the research of methods for their detection. The authors proposed their own model of classification with a description of improvements that increase the efficiency of its work.**

Keywords*: *classification of texts, machine learning, methods of machine learning, logistic regression.

Визначення образ у коротких текстах

Попко Андрій Валентинович
КПІ ім. Ігоря Сікорського
Київ, Україна
andrew.popko.j@gmail.com

Дорогий Ярослав Юрійович
КПІ ім. Ігоря Сікорського
Київ, Україна
cisco.rna@gmail.com

Анотація. **Тези доповіді містять опис вибору математичної моделі для вирішення задачі класифікації коротких текстів за наявність в них образ. В роботі коротко описано дослідження методів їх виявлення. Авторами була запропонована власна модель класифікації з описом вдосконалень, що покращують ефективність її роботи.**

Ключові слова: класифікація текстів, машинне навчання, методи машинного навчання, логістична регресія.

Вступ

Незліченна кількість публічних обговорень різноманітних тем у просторі інтернету багато в чому змінила способи спілкування з іншими. Незалежно від того, чи мова йде про коментарі до статті з сенсаційною новиною, чи про обговорення конкретної відеогри на форумі, сучасні онлайн-простори дозволяють нам легко поділитися власними думками та враженнями, а також отримати теж саме від інших. На зважаючи на те, що часто ці обговорення можуть бути результативними та конструктивними, відносна анонімність, яка приховується в акаунтах користувачів, дозволяє людям залишати образливі та/або неприйнятні коментарі. Ці публікації часто можуть створювати негативне та небажане середовище для інших учасників, що може навіть перешкоджати їм відвідувати сайт. Дана проблема являється достатньо серйозною для адміністрації та власників веб-сайтів.

Одним з потенційно можливих способів пом'якшення цієї проблеми, окрім ручного адміністрування, є створення системи, яка може визначати чи являється кожен вхідний коментар образливим чи ні. За наявності такої власники веб-сайтів матимуть достатньо непогану гнучкість у вирішенні сформульованої вище проблеми. Наприклад, можна вибрати автоматичне блокування точно визначених «образливих» коментарів та приховувати ті, що під сумнівом у системи, для подальшої обробки модератором сайту. Мета цієї роботи полягає в побудові системи, реалізованої на основі методів машинного навчання, яка може класифікувати короткі тексти за ознакою образливості та/чи емоційного забарвлення.

ПОШУК РІШЕННЯ

У вигляді даних для навчання майбутньої системи та тестування було обрано набір з сайту Kaggle [1], який є найпопулярнішим ресурсом та площадкою, де проводяться змагання з машинного навчання. Набір даних містить близько 4000 прикладів, кожний з яких складається з реального тексту, взятого з просторів інтернету та позначки, що свідчить про його забарвленість (1 – образливий, 0 – звичайний). Приблизне співвідношення класифікованих текстів 3/8 відповідно.

Основними показниками оцінки, які використані, були точність тренування (training accuracy) та перехресна перевірка (10-fold cross-validation). Перша

необхідна для визначення наскільки добре мережа працює з вхідними даними та модель оптимізується на навчальному наборі текстів, і в якій мірі вона «перенавчається» на ньому. Точність другої перевірки була більш значущою, оскільки вона визначає наскільки добре модель узагальнюється для тестування на прикладах, що ще не подавались до системи. Це значення є більш точним уявленням про те, як реалізована модель буде функціонувати у реальній ситуації.

Мовою для кодування було обрано Python, що являється найпопулярнішою для реалізації застосувань пов'язаних з нейромережами та машинним навчання. Це дозволило використовувати наявні безкоштовні бібліотеки для підключення, що значно скоротило час на реалізацію алгоритмів. Для виділення характеристик тексту було використано бібліотеку під назвою NLTK (Natural Language Toolkit) [2].

В ході досліджень наявних моделей машинного навчання було визначено що найпопулярнішими являються: Naive Bayes (наївний баєсів класифікатор), SVM (метод опорних векторів), and Logistic Regression (логістична регресія) [3] для класифікації текстів. Спочатку було застосовано кожний з підходів з 1-грамами (unigram), утвореними з вхідних текстів, попередньо підданими обробці (видалення знаків пунктуації, приведення літер до нижнього регістру), а потім вибрано найкращий.

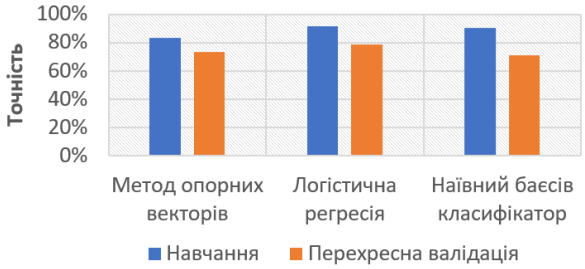

Рис.1. Порівняння базових моделей

Дивлячись на показники для перехресної валідації, очевидно, що модель з логістичною регресією (Logistic Regression) справляється краще ніж дві інші моделі. Тож було вирішено використати даний підхід як базовий для системи класифікації коротких текстів. Також вектор параметрів піддаватиметься навчанню, використовуючи метод стохастичного градієнтного спуску, оскільки він достатньо простий в реалізації.

Для того, щоб підвищити точність моделі були додані деякі функції для створюваного класифікатора. Список тих, що дійсно допомогли покращити показники ідентифікації:

- Стемінг (скорочення слів до основи) слів за допомогою NLTK Lancaster Stemmer;
- Видалення випадкових символів;
- Виділення лексем за допомогою NLTK Tokenizer;
- Використання списку «поганих» слів;
- Використання буквених 4- та 5-грам.

Процес плину показника точності для класифікатора можна побачити на рисунку 2.

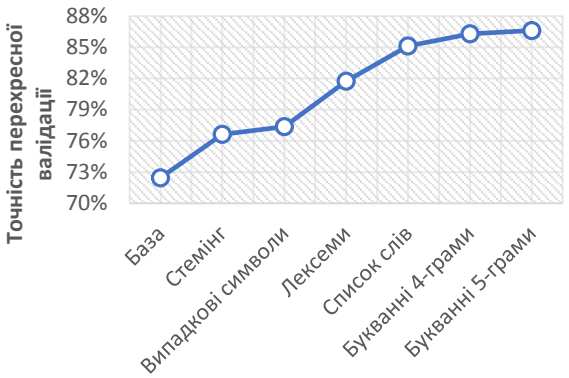

Рис. 2. Покращення моделі в залежності від реалізованих додаткових функцій

Також була спроба замінити використання 1-грам на 2-, 3-, 4-, 5-грами. Крім останніх двох, Помітне покращення спостерігалось тільки при використанні останніх двох функцій.

Завдяки додаванню описаних вище функцій до моделі логістичної регресії система досягла точності навчання 100% на повному наборі даних. В той же час система досягає лише приблизно 86% по точності перехресної перевірки. Отже, стає зрозумілим, що модель значно перенавчилась під час тренування. Два методи, які використані для вирішення цієї проблеми, полягають у наступному:

- Баєсовий апріорний розподіл (Bayesian prior) [4]: замість того, щоб використовувати оцінку максимальної правдоподібності, використано баєсову оцінку максимуму апостеріорної імовірності для визначення ваг. Дані підходи тісно пов'язані, але останній застосовує розширену цільову функцію, що включає апріорний розподіл оцінюваної величини. Це сприяє зменшенню норми вектора параметрів, що дозволяє бути менш чутливим до перенавчання.
- Фільтрація вибору ознак: для кожної з ознак призначено значення, що рівне абсолютній величині синаптичної ваги, визначеній методом стохастичного градієнтного спуску. Ці ознаки будуть більш вагомо впливати на результат під час класифікації. Беручи до уваги лише частину з усіх добутих ознак, величина вектору ознак буде зменшена, що також позитивно впливає на ситуацію з перенавчанням.

Після реалізації вище описаних підходів, точність навчання знизилась до 94%, в той час же показник перехресної валідаціїї дещо збільшився, що свідчить про краще узагальнення отриманих знань.

ПЕРСПЕКТИВИ РОЗВИТКУ ТА ВИСНОВКИ

Завдяки всім перерахованим вище функціям і методам, що закладені в системі, з різними маніпуляціями в налаштуваннях, перехресну точність перевірки вдалось підняти до 86,6% від повного набору даних. Хоча вихідні результати цієї системи є перспективними, існує декілька способів, за допомогою яких можна продовжити покращувати результати. Можна використовувати додаткові характеристики. Даний класифікатор розглядає тексти окремо, проте можна розглядати зв'язок між повідомленнями, наприклад, кількість образливих повідомлень у межах одного потоку повідомлень. На нашу думку, в реальному застосуванні ми також

можемо включити аналіз історії користувача, наприклад, кількість образливих публікацій, які він написав раніше. Також необхідно поекспериментувати, наскільки добре будуть виконуватися інші алгоритми оптимізації, наприклад, метод Ньютона. Ще одним можливим напрямком покращення показників є навчання системи на ідентифікацію сарказмів, коли образа в тексті є прихованою. За допомогою NLP проектів, таких як, наприклад, «Stanford CoreNLP» можна зрозуміти структуру висловлювання та в певній мірі його зміст, що допоможе покращити результати класифікації.

ЛІТЕРАТУРА

1. Kaggle: Your Home for Data Science [Електронний ресурс] : [Веб-сайт]. – Режим доступу: https://www.kaggle.com/ (дата звернення 16.04.2018). – Назва з екрана.

2. Natural Language Toolkit. NLTK 3.2.5 documentation [Електронний ресурс] : [Веб-сайт]. – Режим доступу: https://www.nltk.org/ (дата звернення 16.04.2018). – Назва з екрана.

3. Логистическая регрессия для решения задач классификации [Електронний ресурс] – Режим доступу: https://goo.gl/uu9DsV.

4. Математические методы обучения по прецедентам (теория обучения машин) [Електронний ресурс] / Воронцов К. В. // Курс лекций «Машинное обучение» – С. 18-23. – Режим доступу: https://goo.gl/CoKKBC3. – Назва з екрана.